Calhoun: The NPS Institutional Archive

DSpace Repository

Theses and Dissertations   1. Thesis and Dissertation Collection, all items

1973

Minimizing the cost of projects in Naval shipyards.

Shackelton, Norman John.

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/16703

MINIMIZING THE COST OF PROJECTS IN NAVAL SHIPYARDS

Norman John Shackelton

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

MINIMIZING THE COST OF PROJECTS
IN NAVAL SHIPYARDS

by

Norman John Shackelton, Jr.

Thesis Advisor: W. M. Raike

September 1973

Minimizing the Cost of Projects
in Naval Shipyards

by

Norman John Shackelton, Jr.
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1963
M.S., Naval Postgraduate School, 1971

Submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

from the

NAVAL POSTGRADUATE SCHOOL
September 1973

ABSTRACT

This thesis is concerned with a problem of scheduling that arises in naval shipyards as well as in many other organizations. The problem considered is that of minimizing the total cost of a project with limited manpower available from the various shops and where the number of mandays to accomplish each activity in the project is specified. Total project cost consists of normal direct labor cost, overtime cost, and a penalty for exceeding some specified target date. It is shown that this problem includes several other, more common scheduling problems such as job-shop scheduling. The relationship among the various problems is described including the use of existing solution procedures to solve special cases of the shipyard problem. A mixed integer programming model and a nonlinear programming model are used to describe the system. The mixed integer model consists of several transportation problems linked by precedence relations. An application of dynamic programming to the single shop case of the nonlinear model results in an efficient solution procedure.

TABLE OF CONTENTS

6

LIST OF TABLES

LIST OF DRAWINGS

ACKNOWLEDGEMENT

I. INTRODUCTION

A. THE NAVAL SHIPYARD

The naval shipyard is the industrial activity of the United States Navy. This highly complex organization assembles a wide range of talents and equipment to ensure the operational readiness of the Navy's warships. Shipyard work in support of this mission varies from the single routine repair operation to the complicated network of tasks which is the full scale overhaul. The many different levels of work require many different personnel skills as well as a variety of tools and machines.

Each shipyard is composed of a number of shops whose personnel perform the actual work on the assigned ships. Although the level of skills in each shop may vary, the nature of the type of work to be performed by each shop member is nearly uniform within the shop. Each shop performs some particular class of work such as machining, electrical work, pipe fitting, sheet metal work, etc.

Shipyard management personnel are responsible for conducting several repair operations at the same time. Each ship's overhaul is a project that requires the services of the shop personnel. A limited number of workers are available from each shop so the projects cannot be conducted independently of one another. The common denominator is the shared manpower from the various shops. This restriction on

the use of manpower on projects presents the manager with a difficult problem. Standard PERT/CPM techniques are not usable when resource constraints are present. The Naval Ship Systems Command has developed a management information system called the NAVSHIPS MIS [63] for use in naval shipyards. This system uses PERT/CPM with no resource allocation features to plan and control project schedules. Aggregate workload forecasting is, however, performed to develop estimates on the numbers of the various shop personnel needed to perform the various activities of each project.

B. THE SCHEDULING PROBLEM

The goal of the shipyard commander is to provide as good an overhaul as possible for each ship assigned. This goal must be effected in an atmosphere of severe budgetary restrictions. An additional requirement placed on the commander, therefore, is that of carrying out the mission of the shipyard at as low a cost as possible.

Each project being conducted by the shipyard contributes to total operating cost. Taking a smaller view, each project's total cost is composed of direct labor costs, shop overhead costs, costs for hotel services to the ship, and penalty assessments for exceeding the time alloted for the ship's overhaul.

The variable project costs are the only costs considered here since only these can be lowered. Thus, this thesis considers a project's total cost to consist of direct labor

costs (normal and overtime) and a penalty cost for exceeding a prespecified due date. Labor costs in the shipyard are measured in dollars per manday and the penalty cost is in dollars per day when the project continues past the target date. Hotel services are functions such as garbage collection, steam, water, and electricity and are not related to shipyard personnel assignments. Overtime labor may not be used indiscriminately to reduce project duration. There is a ceiling on the number of overtime mandays that may be expended on any one project.

The scheduling problem faced in the shipyard then, is to minimize the total project cost subject to constraints on the number of various shop personnel employed on the activities that make up the project. Personnel are employed during the normal working day and on overtime, if necessary, to attempt to achieve minimum cost. The total overtime that can be expended is, however, bounded.

This thesis formulates and solves several portions of the shipyard scheduling problem. Chapter II outlines the various problems and solution procedures that make up the theory of scheduling in general. It also shows how the shipyard scheduling problem is related to several other general scheduling problems. This relationship is explored in greater detail in Chapter V where several existing scheduling methods are used to solve some special cases of the shipyard problem.

Chapter III discusses a mixed integer programming model for shipyard project duration minimization and then for the total cost problem. Then Chapter IV uses a nonlinear programming model to solve a similar problem. An application of dynamic programming leads to an efficient solution procedure for the case of a single shipyard shop. Chapter V shows the conditions under which some existing solution procedures can be applied to the shipyard scheduling problem. Finally, Chapter VI gives some ideas for future research in this area of scheduling.

II. RESOURCE ALLOCATION IN PROJECT NETWORKS

Allocating scarce manpower in a naval shipyard so that the total cost of completing a project is minimized is one problem in scheduling that fits into a much larger class. This class of scheduling problems often is given the name resource allocation in project networks. This body of scheduling theory can be broken down into several subclassifications depending on the nature of various scheduling decisions that must be made.

This chapter is devoted to the description of several factors which combine to form this large group of scheduling problems. Each of the classifications is explained and some indications of progress in approaching and solving the various problems are given. Following this, some problems closely related to these are described. The chapter concludes with a description of the shipyard scheduling environment and how it fits into the broader classification. The purpose of organizing this chapter in this manner is to show the interrelationship among several scheduling problems and associated solution methods. A summary of this section is given in Appendix A in the form of a taxonomy of scheduling literature.

A. CLASSIFICATION OF PROBLEMS

Problems in the allocation of resources in project networks can be classified in many different ways. Often, when

characteristics of the shipyard problems coincide with those of other scheduling problems, existing solution techniques can be applied. This is more fully explored in Chapter V.

Different scheduling environments can often be distinguished by the criteria chosen for evaluating an appropriate schedule. Minimum cost, shortest duration and minimum resource expenditure are examples of possible optimizing criteria. Various kinds of resource availability constraints faced by schedulers also generate different types of problems. Additionally, in many different scheduling situations activity characteristics can vary. Some organizations keep track of activity duration while others keep track of the number of manhours or mandays expended on an activity. Combinations of these categories can form together to yield a specific problem type.

1. Optimization Criteria

The criteria aginst which different schedules are evaluated can vary according to an organization's goals. Frequently used criteria are the allocation of resources to minimize total cost or project duration, adjusting a project to achieve level resource expenditures and trading off between cost and time until an acceptable schedule is arrived at.

a. Resource Allocation

(1) Cost Minimization

Many scheduling situations require that an optimal schedule be one that achieves the minimum total cost

of completing the project. The method of computing total project cost may vary depending on the situation. As mentioned in Chapter I, the total project cost in the shipyard consists of direct normal and overtime labor costs along with a penalty for exceeding some prespecified target date. Chapters III and IV of this thesis deal with this criterion for evaluating schedules. Hadley [38, p. 263] uses it in his model formulation. Unfortunately, he suggests no solution procedure.

A more common variation of the total cost criterion includes the cost of changing resource levels. Mason and Moodie [58] have constructed a branch and bound algorithm for minimizing total cost of a project. Three cost terms were involved in their objective function. These costs represented the cost of an increase in resource usage for each activity, the cost of reducing resource level and the cost associated with a change in project duration. This algorithm is applicable only for a single resource. The algorithm, in effect, finds the optimal tradeoff between minimizing project duration and leveling the resource.

A continuous approximation to a problem similar to that approached by Mason and Moodie was used by Cullingford and Prideaux [20]. A cost of changing resource levels and a project duration cost were included in the cost minimization objective. The problems most easily attacked with this procedure are those which have a nearly serial network representation with few activities in parallel.

This permits sequential resource allocation decisions to be made. The continuous approximation can then be made if there are a large number of stages (time periods) associated with completion of the project. Once again, only a single resource is employed on the activities. The problem was formulated as a variational problem and the Euler equations were derived. Several problems with and without resource constraints were examined. The results obtained by applying this method to a project provide planning information in the form of a lower bound on project cost and a suggested project duration.

Another cost objective function included costs of idle and overtime resources, overhead costs related to project duration and costs of changing resource levels. This objective function was included in the SPAR-1 heuristic model developed by Wiest [84]. Wiest's model covered a wide variety of situations and is one of the best scheduling methods that is presently available for very large projects. Wiest has reported [85] that this heuristic method has been programmed in FORTRAN IV to accommodate projects with up to 6000 activities and 25 resource types. Another, easily applied, heuristic procedure is described by Moder and Phillips [61, p. 158].

Each of these papers has been concerned with a cost minimization objective. The nature of activity durations and resource constraints for these problems was not dealt with in this section but is described in later sections pertaining to the appropriate classifications.

(2) Duration Minimization

Finding a schedule that completes a project in the shortest possible time is the goal of the vast majority of scheduling methods. The methods discussed in this section all have in common the duration minimization objective but differ in their types of resource constraints and other characteristics. These other aspects are discussed in later sections of this chapter.

The goal of minimizing project duration is related to the cost minimization objective in that each of the expressions for total project cost include some function of project length. In some cases this consists of a penalty cost and in others simply a cost associated with lengthening or shortening the project.

Among the earliest of the methods for minimizing project duration were the PERT and critical path techniques [45], [56]. Since these methods did not take scarce resources into account, many authors have attempted to extend these results to include the restrictions on resources.

The earliest successful results employed heuristic rules for scheduling activities. Kelley [47] described several methods for ordering activities and then selecting them for a schedule. Although his methods did not guarantee optimality, they did provide a feasible solution to a large combinatorial problem. Some modifications to Kelley's methods were later made by Patton [67]. Patton constructed a branch and bound algorithm which guaranteed

an optimal schedule for small projects. Lambourn [49] described a proprietary system called RAMPS which used heuristics to solve a wide variety of scheduling problems including the problem of minimizing project duration.

Wiest [83] described some properties of schedules which were later used as a basis for heuristic rules in his SPAR-1 model [84]. He defined a "critical sequence" of activities that resembled the critical path in unconstrained resource procedures. Activities not in this critical sequence could be shifted to other positions in the schedule much like slack activities in the critical path method. Wiest's methods for attacking constrained resource problems are briefly described in Reference 85.

Implicit enumeration procedures have been used to solve a wide variety of combinatorial problems. Two applications of branch and bound procedures that have received notoriety are Land and Doig's solution to integer programming problems [50] and the traveling salesman algorithm of Little and others [55]. Because of the combinatorial nature of resource constrained scheduling problems, an implicit enumeration technique is a logical choice to arrive at a solution method. The earliest applications of branch and bound to scheduling were in the area of job shop sequencing. These methods are included in Section B of this chapter.

Fisher [29] formulated the job-shop scheduling problem as a zero-one program. All the variables in the constraints of his formulation have coefficients which

are zero or one. Fisher developed a column generation procedure for solving the problem in this form. When a more general network scheduling problem is formulated, the coefficients are no longer zero or one and the column generation procedure is no longer applicable. Fisher proposed an implicit enumeration procedure for this case.

Johnson [43] developed a branch and bound method for minimizing the duration of a small project with constraints on resource availability. In his thesis, Johnson assumed that once an activity had begun, the resources assigned to it remained fixed throughout the duration of the activity. This assumption was also made by Schrage [76] in his branch and bound method for minimizing project duration. Schrage's approach is discussed in somewhat more detail in Chapter V. This assumption was relaxed by Patton in his doctoral dissertation [67]. His branch and bound procedure allowed activities to exhibit a quality that Conway, Maxwell, and Miller [19, p. 169] call preemptive resume. That is, activities may be scheduled for a time, interrupted and then resumed when possible. Each activity is completed when its total duration equals some prescribed length of time. Patton's thesis presents an interesting method for enumerating all the schedules that need be evaluated. Preemptive resume is also allowed by Schrage [77] in a modification to his earlier procedure. None of these methods will accommodate very large projects but do provide a means for evaluating heuristics and for providing insight into scheduling problems.

The use of disjunctive graphs to represent scheduling problems has been investigated by several authors. Balas [2], [3], Gorenstein [3], and Raimond [70], [71] have each formulated the problem of minimizing project duration subject to resource constraints using disjunctive graphs. Disjunctive graphs and their application to the shipyard scheduling problem are discussed in Chapter V. Each of the above authors has represented his disjunctive graph formulation by a mixed integer program. The differences in the methods devised by the authors arise from the different integer programming solutions. Balas used the Benders partitioning procedure [7] for the mixed integer problem and his additive algorithm [1] to solve a zero-one programming problem then generated. His method consists of solving an alternating sequence of simple critical path problems and zero-one programming problems. Raimond used a direct search algorithm of Lemke and Spielberg [53] to solve the mixed integer programming problem. Gorenstein's method is an implicit enumeration solution to a mixed integer programming problem. Although each of these methods solves only small problems, the use of disjunctive graphs in solving scheduling problems should be promising in the future.

An assembly line balancing algorithm developed by Gutjahr and Nemhauser [36] provided an analogy to the resource constrained scheduling problem for Davis and Heidorn [26]. Each activity in their method is broken up into a series of tasks each requiring a single unit of time

for accomplishment. All feasible ways of scheduling activities in each time interval are generated and a shortest path algorithm is then used to find the schedule with the shortest duration. The article reports that the algorithm has been programmed to accommodate up to 220 tasks and 5 resource types. This severely limits the lengths of activities that may be present in the original network.

The interest in three other duration minimization problems [9], [73], and [74] lies in the nature of resource constraints and activity characteristics and are discussed in the appropriate sections.

b. Resource Leveling

The usual objective of resource leveling procedures is to assign men from various shops in order to achieve near constant resource allocations over time. A due date for the project is normally fixed in advance and resource allocations must then be made over this fixed horizon.

The earliest attempts at devising methods for achieving constant workloads were heuristic in nature. Burgess and Killebrew [14] described the relationship between arrow diagrams and Gantt charts and showed how to graphically shift jobs (activities) until a constant resource allocation over time was achieved. In their paper, computer programs were presented which generated a manpower loading chart and minimized the variation in resource allocation throughout the course of the project. A modification to this procedure appears in the book by Moder and Phillips

23

[61, p. 163]. At nearly the same time Levy and others [54] developed a system for smoothing manpower requirements for shops in naval shipyards. Their procedure attempts to reduce the peak manpower requirements by first scheduling jobs in accordance with their earliest possible starting times and then moving appropriate jobs to slack periods. This is done by fixing a "trigger level" at one resource unit below the peak resource requirement for each shop. An attempt is then made to schedule all jobs so that the activity manpower requirements are below this trigger level.

A few years later Moodie and Mandeville [62] showed a relationship between resource leveling and assembly line balancing. They adapted an integer programming model of Bowman [12] to the resource smoothing problem and were able to solve some very small problems. The method proved to be impractical for large networks. This connection between resource leveling and assembly line balancing led Davis and Heidorn [26] to their solution procedure for minimizing project duration.

Razumikhin [72] addressed a resource leveling problem where a specified number of mandays to perform each activity was given. He solved the problem by minimizing the mean square deviation of resources expended from a constant. Razumikhin's method was interesting in that a hydrostatic model was used to model the situation. The potential energy of the fluid mechanic system corresponded to the mean square deviation of resource level from a constant. A method of

successive approximations was used to solve it. In a later paper, Razumikhin [73] approached the same problem, this time using a nonlinear programming model. An algorithm exhibiting monotone convergence was given for solving the nonlinear programming problem.

Beale's method of quadratic programming [6] was applied by Voronov and Petrushinin [81] to a resource leveling problem. In their problem, each of the event times were assumed fixed in advance and a fixed number of mandays was required for each activity's completion. The objective used was the minimization of the squared difference between resources expended and a constant. The assumption of fixed event times was also required by Petrovic [68] in his solution to the resource leveling problem. He formulated the problem as a multistage decision problem and utilized dynamic programming for its solution. The large amount of computation involved caused Petrovic to suggest several successive approximation techniques for solving the problem. The objective function for the resource leveling problem was again quadratic. Additionally, Petrovic mentioned several other resource allocation problems that could, in principle, be solved by this method. The primary targe of his paper, however, was the leveling of expended resources.

c. Time/Cost Tradeoff

Time/Cost tradeoff procedures are those which arrive at a schedule for a project network which has the "best" balance between project duration and total resource

25

expenditure (usually dollar cost). Most methods of this class involve no restrictions on resources and the relative value between duration and cost is determined by management.

Several authors have developed procedures for solving problems of this type when each activity has a linear relationship between cost and duration. These procedures have been treated extensively elsewhere so are not discussed here. Some of the papers concerned with this problem are References [18], [46], [79], [30], and [17]. Some modifications to the original linearity assumptions have been made to correspond to other scheduling environments.

Berman [8] investigated the time/cost tradeoff problem where activity cost was a convex function of activity duration. An iteration procedure was given for balancing a network so that for each event the sum of the slopes of the cost functions for all activities leading into the event equalled the sum of slopes for all activities leading out of the event. It was shown that when the network is balanced in this manner, for fixed duration, total project cost is minimized. A convex programming algorithm was used by Lamberson and Hocking [48] in solving a similar problem. The purpose of their procedure was to reduce the duration of a project by allocating additional resources along the critical path. Each activity had associated with it a cost which was a convex function of time.

A branch and bound algorithm for finding the minimum total project cost for concave activity cost-time

curves was developed by Falk and Horowitz [27]. Their procedure yields the optimal cost by successively underestimating by linear interpolation the concave cost-time functions. At each step the activity with the "worst underestimate" is chosen as a branching variable and underestimation at the next stage is performed with two line segments. Each iteration produces an upper bound which is lower than all previous stages and a lower bound. Branching on the lowest current lower bound, the algorithm terminates when upper and lower bounds are equal.

The presence of constraints on resources in the time/cost tradeoff problem was investigated by Nikonov and Pluzhnikov [65]. This paper presented an algorithm for constructing the project cost curve when a constraint on the number of resources (other than cost) used between event times was added. The resource constraint considered by Nikonov and Pluzhnikov was piecewise linear. The algorithm employed was an extension of that developed by Kelley [46].

Jewell [42] presented a solution for a critical path problem in which some activities do not have a unique location in the network. These activities may be moved to other locations or divided into smaller tasks and these subtasks moved to various specified locations in the network. The Dantzig-Wolfe decomposition procedure [22], [23] was applied to the divisible activities case to provide a solution method. A branch and bound solution to an integer programming formulation of the movable activities case was given in the paper.

2. Nature of Resource Constraints

A wide variety of restrictions can be imposed on the quantity of resources that are available to a manager for employment on a project. The number of available resources at any time may be specified by a step function of time or resource profile. Another way is specifying some fixed total number of resources that can be made available for a project. Each of these types of resource restrictions generates a different problem for the scheduler.

a. Resource Availability Profiles

Resource availability profiles are step functions which represent the number of resources of a certain type that are available at any given instant during the course of a project. There are three types of resource availability profiles: fixed, variable, and constant.

(1) Fixed Resource Availability

A fixed resource availability profile can be represented by a step function similar to that in Figure 1. This is a fixed resource profile if the times τ_1, τ_2,..., τ_q at which changes in resource availability occur are fixed and known in advance by the scheduler. R_t^k represents the number of resources of type k available at time t. This number is an upper bound on the quantity of resources of that type that can be employed on any activity in the network at time t.
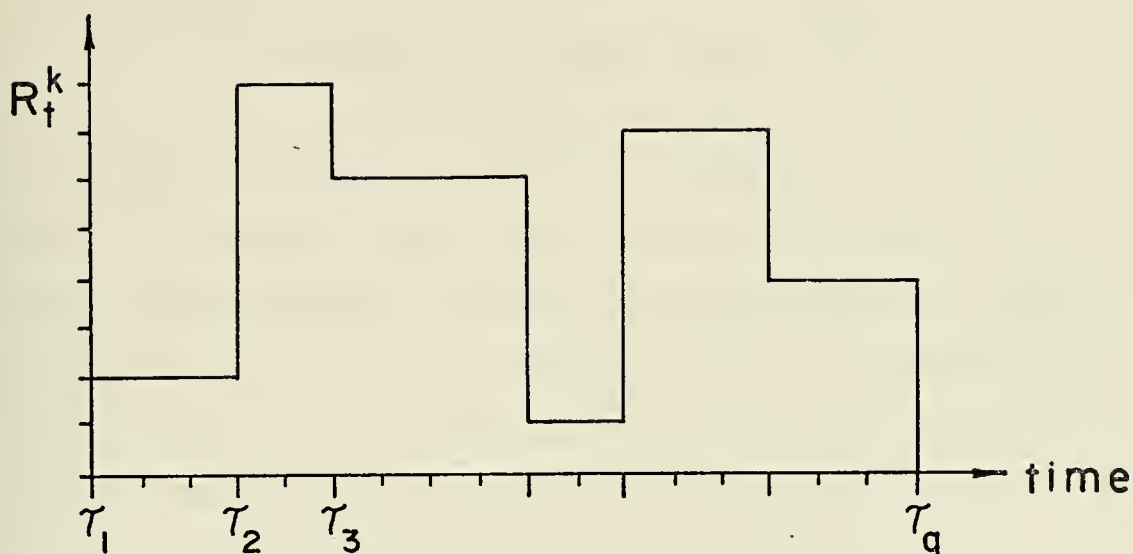
Figure 1

This is the nature of the restriction on
the use of various shop personnel in the shipyard scheduling
problem. Fixed resource availability profiles are used in
the mixed integer programming formulation of the shipyard
problem in Chapter III of this thesis and again in Chapter
V.

This type of resource constraint is not only
applicable to the naval shipyard, however. This is the form
of the resource constraints that are present in Wiest's
SPAR-1 [84] and Lambourn's RAMPS [49]. Karush [44] developed
a method for enumerating all schedules necessary for evalua-
tion when resources are constrained by a fixed resource

availability profile and each activity exhibits a nonincreas-
ing resource usage profile. This term is defined in a later
section in this chapter.

(2) Variable Resource Availability

A profile representing variable resource
availability is shown in Figure 2. Note that the break-
points of the step function are numbered t_1, t_2, ..., t_N.
These times represent the node or event attainment times of
the project and are thus variable. $R_{t_i}^k$ then represents the
number of resources of type k available between events i
and i+1. Henceforth, this will be represented by R_i^k.

Variable resource availability profiles do
not have much physical meaning in themselves but they can be
used as an approximation to the fixed resource availability
profile.

This thesis uses this approximation in
Chapter IV to obtain an efficient solution procedure for the
shipyard scheduling problem. This method of constraining
resources was also used by Nikonov and Pluzhnikov [65] in
a resource constrained time/cost tradeoff problem.
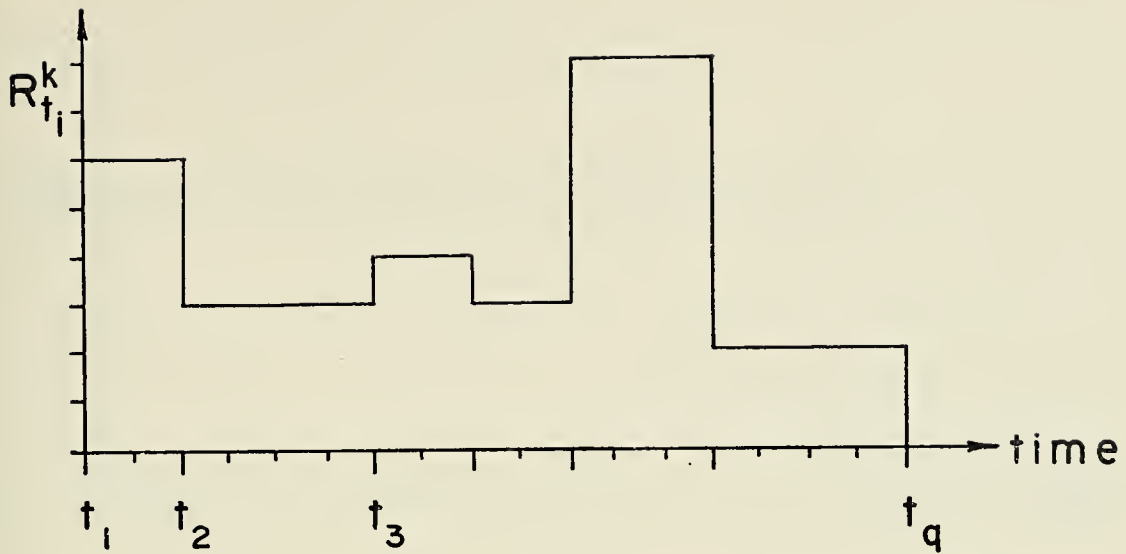
Figure 2

(3) Constant Resource Availability

Figure 3 is a graphical representation of
a constant resource availability profile. A fixed quantity
of resources is available at every time point throughout the
project. R^k represents the maximum number of resources that
can be utilized by any activity of the project.

Because of its simplicity, this type of
resource constraint is the most often used in scheduling
procedures. This form of resource constraint was employed
in references 2, 3, 26, 34, 47, 67, 70, 71, 76, and 77.
These papers were briefly described in the section on opti-
mization criteria and are not discussed further here.
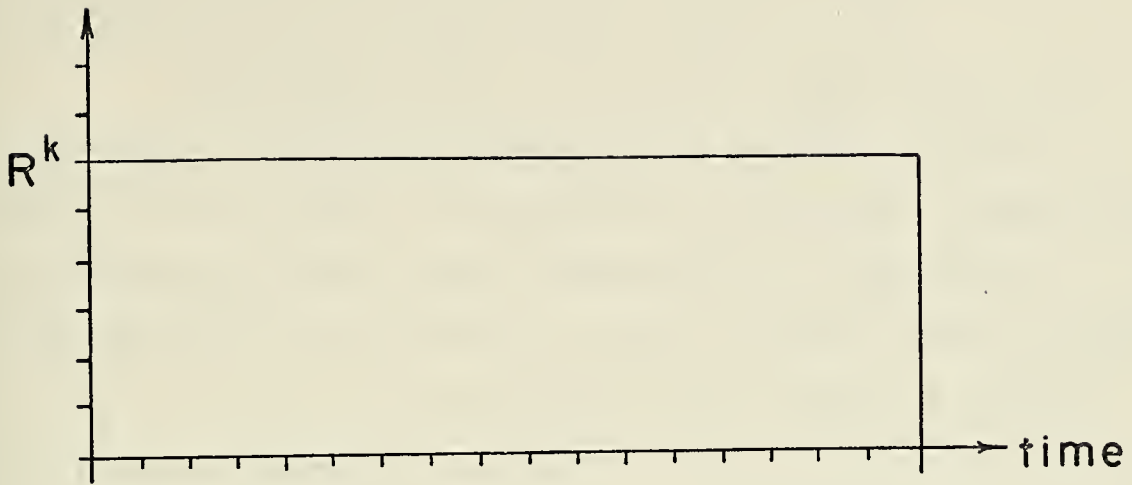
Figure 3

b. Bounded Resource Volume

A constraint in the form of a restriction on the
total volume of a resource that can be utilized throughout
a project has sometimes been imposed in certain scheduling
situations. A resource type that could fit into this cate-
gory is some construction material that once used is no
longer available.

Bershchanskii [9] solved the problem of minimizing
project duration subject to bounds on total resources of
various types employed. Each activity's duration was a convex
function of the resources employed on the activity. Lagrange
multipliers were introduced and a saddle point analysis of
the Lagrangian was performed. This led to an iterative
procedure based on Rosen's gradient projection method [75].

If the activity duration function is linear Bershchanskii
shows that Kelley's procedure [46] could be used.

Razumikhin [73], [74] gave two approaches for
the problem of minimum duration with a bounded resource
volume. One of these [74] developed a hydrostatic model for
the scheduling system. The other paper [73] required a
strict ordering of the events so that the event times satisfy
$t_1 < t_2 < \cdots < t_n$. The problem in this case was solved
as a linear programming problem.

3. <u>Activity Characteristics</u>

a. Activity Duration Estimate

The most common estimate associated with an
activity in project scheduling is a duration estimate.
Most of the references mentioned in the previous two sections
require that each activity have a fixed duration associated
with it. Then corresponding to this fixed duration is a
level of resources required to complete the activity within
that time period.

b. Fixed Resource-Time Unit Requirements

Some schedules require that a fixed number of
resource-time units be expended in order that an activity be
completed. Examples of some often used resource-time units
are manhours, mandays, and machine-hours. In naval shipyards
the normal unit for measuring work is the manday and this
unit is used throughout this thesis.

Mason and Moodie [58] associated a fixed number
of resource-time units with each activity in their branch

33

and bound algorithm for cost minimization. Cullingford and Prideaux [20] adopted a similar method for total activity work performed. Fixed resource-time units were required by Bershchanskii [9] and Razumikhin [73], [74] in their duration minimization procedures. Finally, Petrovic [68], Razumikhin [72], [73], and Voronov and Petrushinin [81] each used manday estimates in leveling manpower.

c. Resource Usage Profiles

Another way to represent activity accomplishment is with a resource usage profile. An example of a resource usage profile is shown in Figure 4 for an activity (i,j) with start event i and total duration T_{ij}. The number of resources of type k employed at time t on activity (i,j) is given by $r_{ij}^k(t)$.
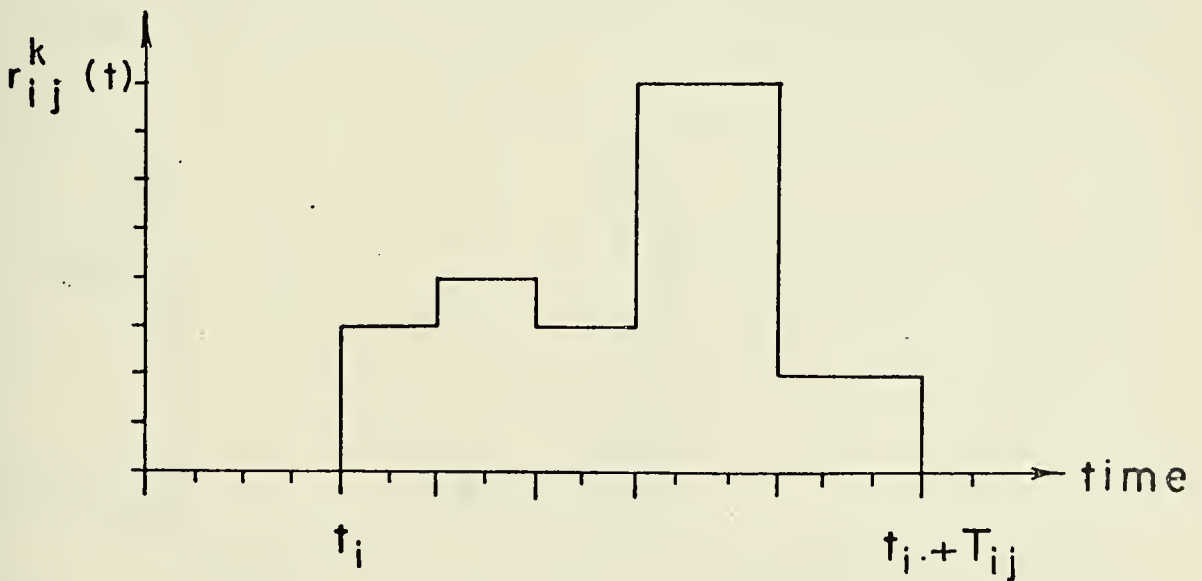
Figure 4

Karush [44] coined the term activity resource profile and gave conditions under which minimum duration schedules could be generated for a project whose activities had resource profiles associated with them. This is further discussed in Chapter V.

d. Time/Cost Tradeoffs

In the critical path approach [30] there is associated with each activity a linear cost function of the form shown in Figure 5(a). This gives the possible durations T_{ij} that the activity (i,j) can assume and the cost associated with those durations. Figure 5(b) illustrates a similar case where the possible durations and associated costs are discrete. Moder and Phillips [61, p. 196] presented a procedure for calculating minimum project duration where discrete time/cost tradeoff points were associated with each activity. The solution so generated is not guaranteed to be optimal.
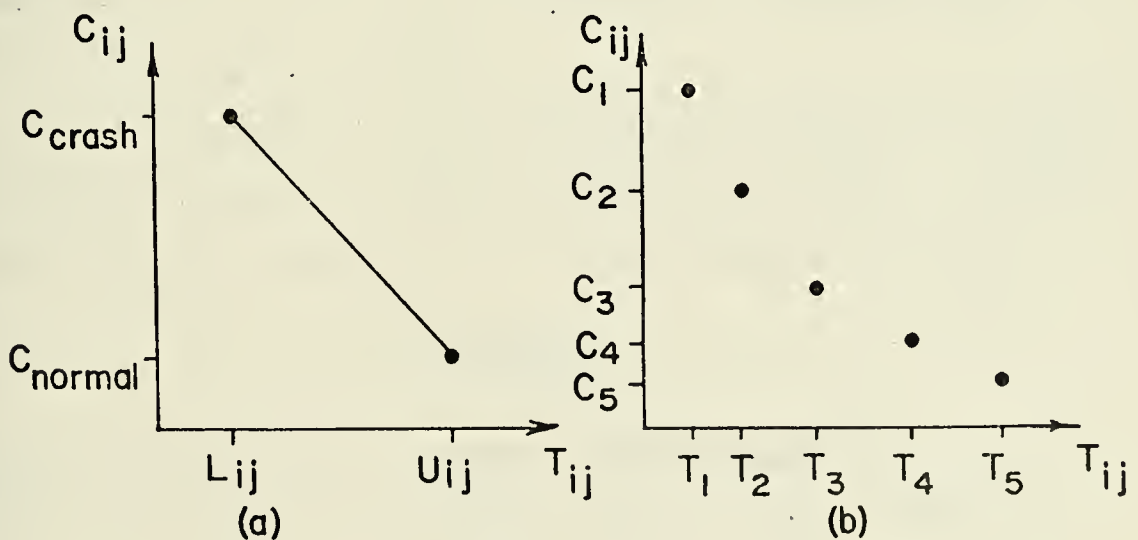
Figure 5

B. RELATED PROBLEMS

A number of frequently encountered problems in the theory
and application of scheduling are closely related to the net-
work-based problems described above. Some problems in sched-
uling that are similar to resource allocation in projects
are job-shop scheduling, flow-shop scheduling, assembly line
balancing, and multiproject scheduling. The relationships
between these problems and resource allocation problems are
briefly described below.

1. Job-Shop Scheduling

Job-shop scheduling problems are often phrased:
"Determine the order of processing of n jobs on m machines
so that the time to process all jobs is minimized. A fixed
ordering of machines for each job is specified in advance."
This type of problem can be represented by a project network
with a constraint on resources. If a machine can only pro-
cess one job at a time then the resource availability is
one for that type of machine.

Consider the following example: Minimize the time to
process all jobs on all machines where the machine orderings
and processing times for each job are given in Table I.

Table I

Job	Machine (Process Time)			
1	A(3)	B(3)	C(7)	D(6)
2	B(5)	A(6)	D(2)	C(2)
3	A(4)	C(2)	D(3)	B(4)

This problem can be represented by the project network of Figure 6. Each activity represents the operation of processing a job by a machine. Each chain from node 1 to node N represents a job. The resources of type k (k stands for machine A, B, C, or D in this case) employed on each activity must not exceed 1. A generalization of this is created when the paths from source to sink are not required to be parallel. The job shop scheduling problem with a general network relationship between operations is called, in this thesis, the network job shop scheduling problem. A more complete examination of the relationship between job-shop scheduling and resource allocation in project networks is made in Chapter V.
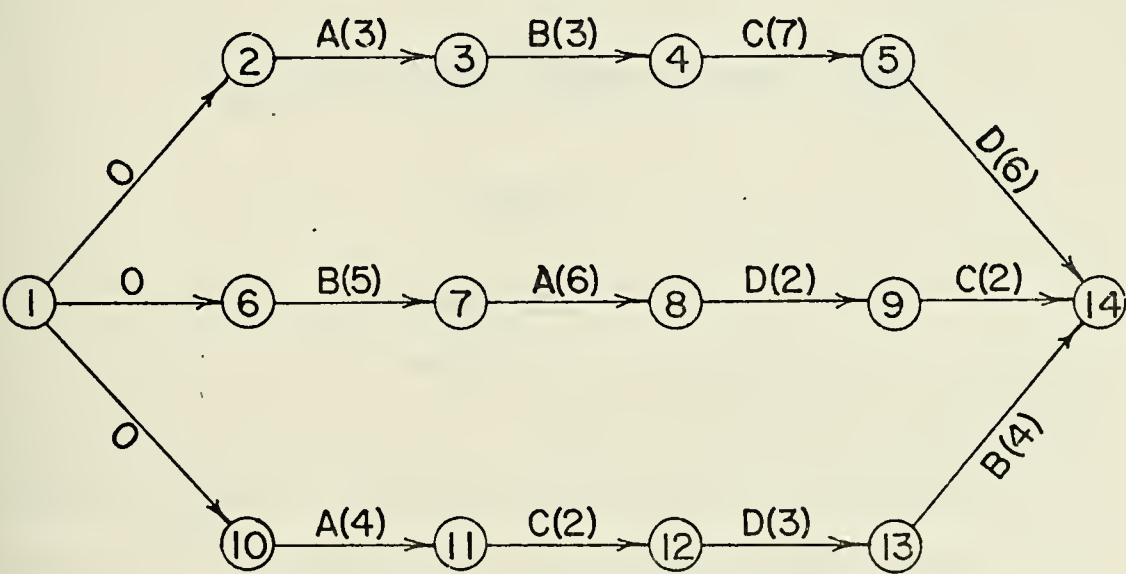
Figure 6

2. Flow-Shop Scheduling

Flow-shop scheduling is a special case of job shop scheduling where each job has the same order of machines associated with it. Suppose the previous example had the machine ordering A, B, C, D for each job. This situation can then be illustrated by Figure 7. An identical resource constraint is imposed.
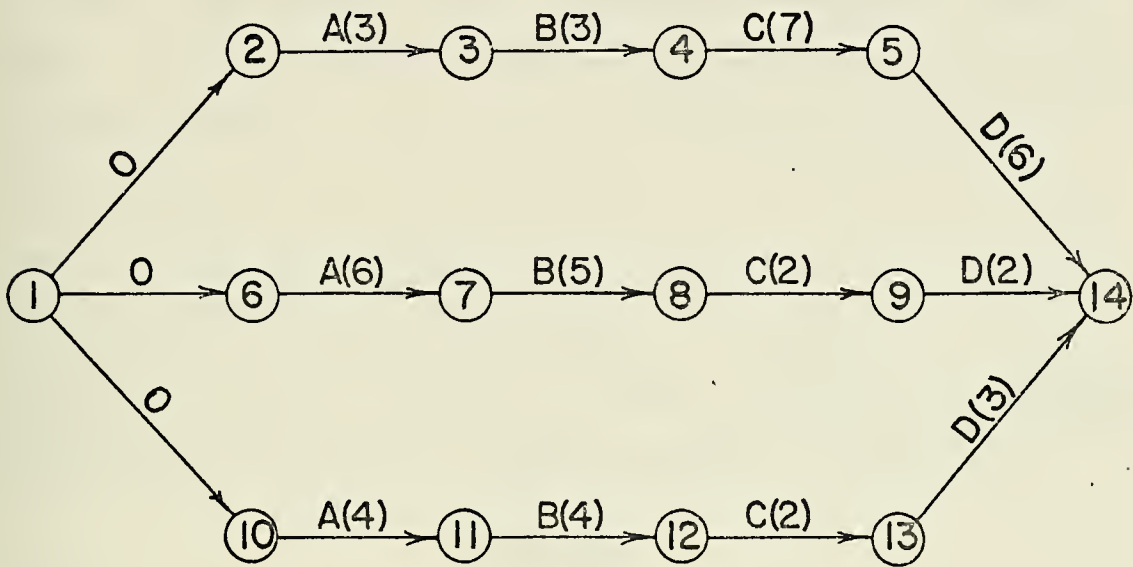
Figure 7

3. Assembly Line Balancing

An excellent comparison of assembly line balancing with resource constrained projects was made by Moodie and Mandeville [62]. An assembly line can be represented by a network in which the nodes are operations that must be performed in order that the product be assembled. The arcs of the network ensure that the operations are performed in the correct technological order. A work station is assigned to perform one or more of the operations in the network. It

38

is usually assumed that each operation requires some fixed length of time to accomplish. As more operations are assigned to a work station, the longer the product to be assembled is held in the work station. The cycle time is the largest processing time of all the work stations and is the time between assemblies of the product. Two different optimization criteria are often used to balance assembly lines [19, p. 140]. One of these is to minimize the number of work stations while each work station processing time cannot exceed some constant. The other is to assign operations to work stations so that the maximum work station processing time is a minimum. Each of the parameters of the assembly line balancing problem corresponds to a parameter of the constrained resource problem. Moodie and Mandeville compared the problem entities in a manner similar to Table II.

Table II

Assembly Line Balancing Problem	Resource Allocation in Project Networks
Operation	Activity
Operation Processing Time	Activity Resource Level
Number of Work Stations	Activity Duration (Discrete Units)
Cycle Time	The Largest Activity Resource Level

The two assembly line balancing problems are analogous to two problems of allocating scarce resources in

projects. The first of these is to minimize the time to complete all activities while maintaining all activity resource levels below some constant. The other problem is to schedule activities in time periods so that the maximum activity resource level is minimized. This is the resource leveling problem. This clear-cut correspondence between the two problems was used by Moodie and Mandeville to describe the resource leveling problem and by Davis and Heidorn [26] to solve the resource constrained duration minimization problem. A review of the literature of assembly line balancing up to 1965 was presented by Ignall [40].

4. Multiproject Scheduling

Coordinating an organization's many individual projects is the goal of a multiproject scheduling system. Usually, independent project networks can be connected by dummy activities forming a giant project network. Then, conceptually, any of the objectives and procedures can be applied to the entire multiproject network. The large size of such a network makes this procedure infeasible. In addition, each project composing the multiproject system may have a due date. Special procedures are needed to handle such problems.

The heuristic multiproject system RAMPS (Resource Allocation and Multi-Project Scheduling) [49] was designed to perform several analyses of multiproject networks. The RAMPS system is able to handle 700 activities, 60 resources and 6 projects each with their own desired due dates. Restrictions on the resources such as those described in previous

40

sections are included in the package, (available only from CEIR, Inc.). Parikh and Jewell [66] presented a procedure for analyzing a large network which could be decomposed into several subprojects. Associated with each sub-project was a piecewise linear project cost curve. By considering each subproject as a single activity with a piece-wise linear time/cost tradeoff curve, standard time/cost tradeoff procedures could be used. This is nearly parallel to the multi-project system where there are no resource restrictions and each project's cost curve has been determined.

Pritsker, Watters, and Wolfe [69] formulated a zero-one programming problem for the multiproject scheduling problem. The program's versatility allows the inclusion of a wide variety of objectives but, because of the combinatorial nature of the problem, can only be used for very small sub-projects.

C. THE NAVAL SHIPYARD SCHEDULING PROBLEM

The shipyard scheduling environment is similar to that of any multiproject organization. The overhaul or repair operations performed on a single ship provide the activities and events for a single project. Each of the projects share fixed resources and each contributes to the total cost of operating the shipyard. A typical naval shipyard has several of these projects going on at the same time. Each of the ships must be returned to its operational duties at or near some prespecified date. The shipyard must then attempt to

41

perform the necessary work on all of its assigned ships within some strict time frame at the lowest possible cost. It is further required that each ship's work package be completed as close as possible to the specified departure date. A high cost is assessed for failure to meet a ship's departure date.

Fixed numbers of personnel in each of the shipyard shops place additional difficulties on project scheduling. Personnel may not be hired or fired simply because an additional project is accepted or a project is dropped by the shipyard. Therefore, when some project is accepted by the shipyard, each shop can make manpower available in the form of a fixed resource profile such as shown in Figure 1.

Each element of the work package is an activity in the project network that represents the entire work package. Some specified amount of work must be performed to complete each activity. Most operations have been performed many times in the past so each shop foreman can, based on experience, make a good estimate of the number of mandays required to perform the operation.

The scope of this dissertation is limited to the single project case. It is assumed here that a project is faced by a fixed resource profile and cannot borrow resources from other projects (except during overtime). It is also assumed that if the total cost of each single project is minimized, then so is the shipyard operation cost. This assumption is valid for fixed assignments of shop personnel to projects.

42

This is the situation that faces a shipyard when an additional ship is assigned to the shipyard for repair after most resources have been committed for other projects. The idle resources on all other projects then form the resource profiles for the various shops.

III. MIXED INTEGER PROGRAMMING MODEL

A. INTRODUCTION

Integer programming models provide means of expressing complex problems that are not easily formulated using other mathematical models. Since the appearance of Wagner's integer programming model of the job-shop [81] many authors have used integer programming to describe their scheduling problems. Some of these papers have concentrated on job shop type problems [11], [21], [29], [35], [57], and [80], while others have been concerned with more complex scheduling problems such as assembly line balancing [62], project scheduling [38], [67], and multiproject scheduling [69]. The models contributed by these papers have provided a foundation for scheduling theory and have often led to efficient solution methods for specific problems.

In this chapter, the problem of minimizing total cost of a shipyard project with restrictions on the available resources is formulated as a mixed integer-linear programming problem. The formulation is begun by characterizing the minimization of project duration. This is then extended to the cost minimization problem.

B. ASSUMPTIONS AND NOTATION

A project is composed of m activities related by the logical sequences in which they must be performed. The logical relationships among the activities are the same as

44

the precedence relations in PERT/CPM. The ordered pair (i,j) represents the activity starting at event (node) i and terminating at event j. There are n events in the project and N represents the collection of activities and events making up the project. Each activity (i,j) in the project requires a fixed number of mandays M_{ij} for completion. There are K shops available and each activity utilizes men from exactly one shop. It is assumed that the project can be completed within T days. On day t, shop k, k = 1, 2, ..., K, has a limited number of men available given by R_t^k, t = 1, 2, ..., T. These men may be utilized by any activities that are permitted by the precedence relations to be in progress on day t and that require men from that particular shop.

Let r_{ijt}^k be the number of men from shop k employed on day t in the performance of activity (i,j). Then the number of men utilized over the entire project on activity (i,j) is

$$\sum_{t=1}^{T} r_{ijt}^k \; ; \quad (i,j) \; \varepsilon \; S_k, \quad k = 1, 2, \ldots, K.$$

Here S_k is the set of all activities requiring shop k manpower. Then the total number of men from shop k employed on day t is

$$\sum_{(i,j) \varepsilon S_k} r_{ijt}^k \; ; \quad k = 1, 2, \ldots, K, \quad t = 1, 2, \ldots, T.$$

Two assumptions that must hold for the mixed integer programming model to be valid are:

1. The network precedence relations among the activities must be maintained.

2. Activities may not be separated and are an integer number of days in length.

Assumption 1 means that no men may be employed on activity (i,j) until all activities (k,i) are completed. Assumption 2 requires that once men are assigned to an activity then there must be nonzero manpower assignments to that activity until it is completed. This second assumption is relaxed in Section III-D-3.

C. MINIMIZATION OF PROJECT DURATION

The first task is to minimize the duration of a project while ensuring that no more than the available numbers of men are used, the activity precedence relations are not violated, and that the required number of mandays to complete each activity is expended.

1. Transportation Problem Representation

The first sets of constraints can be developed by noting that the total number of men from shop k employed on day t must not exceed the number of men from that shop that are available on that particular day. This is represented by

$$\text{(III-C-1)} \qquad \Sigma_{(i,j)\varepsilon S_k} \; r^k_{ijt} \leq R^k_t \; ; \quad \begin{array}{l} k = 1, 2, \ldots, K \\ t = 1, 2, \ldots, T. \end{array}$$

In addition, the total number of mandays expended on activity (i,j) must equal the number of mandays required to complete

the activity. This is

$$\text{(III-C-2)} \quad \sum_{t=1}^{T} r_{ijt}^{k} = M_{ij} \; ; \; (i,j) \; \epsilon \; S_{k}, \quad k = 1, 2, \ldots, K.$$

It is also required that the manpower allocations be nonnega-
tive. The inequality (III-C-1) and the equation (III-C-2)
together are, for fixed k, in the same form as the constraints
of the transportation problem with inequalities [37, p. 312].
Each R_{t}^{k}, t = 1, 2, ..., T can be viewed as the availability
at an origin and each M_{ij}, for every (i,j) ε N is the demand
at a destination. With every shop k, k = 1, 2, ..., K
there is then associated a set of transportation problem con-
straints with T origins and m_{k} destinations. Here m_{k} is the
number of activities in the set S_{k}. The right-hand side of
the constraints (III-C-1) can be represented graphically by
a resource profile similar to that of Figure 1 of the previous
chapter.

        If the time to complete the project is to be mini-
mized, the earlier time periods should be made more attrac-
tive for resource allocations than the later ones. To do
this a low cost could be assigned to the first time period
manpower allocation for each activity, a larger cost for the
second allocation for each activity and so forth until each
possible allocation had a cost associated with it. The
objective would then be to minimize total cost. This is

(III-C-3)   Minimize $\displaystyle\sum_{k=1}^{K}\sum_{t=1}^{T}\sum_{(i,j)\epsilon S_k} c'_t r^k_{ijt}$

with $c'_1 < c'_2 < \ldots < c'_T$. An appropriate assignment of costs might be to put $c'_1 = g^1$, $c'_2 = g^2$, $\ldots$, $c'_T = g^T$ where g is some positive constant larger than 1.

Using a method described by Hadley [37, p. 312] (III-C-1) and (III-C-2) can be transformed into equations. The method corresponds to simply adding dummy destinations representing the total slack added in (III-C-1). This then becomes the K separate transportation problems

Minimize $\displaystyle\sum_{k=1}^{K}\sum_{t=1}^{T}\sum_{(i,j)\epsilon S_k} c'_t r^k_{ijt}$

Subject to

(III-C-4) $\displaystyle\sum_{(i,j)\epsilon S^*_k} r^k_{ijt} = R^k_t$; $\begin{array}{l} t = 1, \ldots, T \\ k = 1, \ldots, K \end{array}$

$\displaystyle\sum_{t=1}^{T} r^k_{ijt} = M_{ij}$; $(i,j)\epsilon S^*_k$, $k = 1, \ldots, K$

$r^k_{ijt} \geq 0$; $\forall\, i, j, k, t.$

S^*_k is the set S_k with the addition of slack variables. This representation does not, of course, take into account the precedence relations among activities.

It is now convenient to place (III-C-4) in matrix form. Let

$$c_k^! = (c_1^!, c_2^!, \ldots, c_T^!), \quad k = 1, \ldots, K \quad \text{and}$$

$$x_k = [r_{\alpha 1}^k, r_{\alpha 2}^k, \ldots, r_{\alpha T}^k, \ldots, r_{\beta 1}^k, \ldots, r_{\beta T}^k], \quad k = 1, \ldots, K.$$

Here $(\cdot)$ represents a row vector and $[\cdot]$ denotes a column vector. Additionally $\alpha, \ldots, \beta$ represent activities and $\{\alpha, \ldots, \beta\} = S_k^*$. Also let

$$A_k x_k = \left|\left| \Sigma_{(i,j) \epsilon S_k^*} r_{ijt}^k ; \quad t = 1, \ldots, T \right|\right|$$

where $||\cdot||$ denotes a matrix. Also

$$B_k x_k = \left|\left| \sum_{t=1}^{T} r_{ijt}^k ; (i,j) \epsilon S_k^* \right|\right|,$$

$$a_k = [R_1^k, \ldots, R_T^k],$$

and

$$b_k = [M_\alpha, \ldots, M_\beta].$$

The K transportation problems are then represented by

Minimize $c_1 x_1 + c_2 x_2 + \ldots + c_K x_K$

delete

Subject to

$$A_1 x_1 \qquad\qquad = a_1$$

$$B_1 x_1 \qquad\qquad = b_1$$

(III-C-5)
$$A_2 x_2 \qquad\qquad = a_2$$

$$B_2 x_2 \qquad\qquad = b_2$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$A_K x_K = a_K$$

$$B_K x_K = b_K$$

$$x_1,\ x_2,\ \ldots,\ x_K \geq 0$$

Constraints indicating the precedence of the activities must now be added.

2. Precedence Relationships

In order that the logical sequence of activities be preserved, certain relationships concerning the allocations r_{ijt}^{k} must hold. If an activity (i,j) must immediately precede another activity (j,p), then for any time period t in which $r_{ijt}^{k} > 0$ then $r_{jp1}^{q} = \ldots = r_{j,p,t}^{q} = 0$. The superscript q simply shows that (j,p) may use a different shop. Also, activities may not be separated so for some activity (i,j), if $r_{i,j,t-1}^{k} > 0$ and $r_{ijt}^{k} = 0$ then $r_{i,j,t+1}^{k} = \ldots = r_{ijT}^{k} = 0$.

Finally, if activity (j,p) is preceded by activity (i,j), then (j,p) cannot begin until

$$\sum_{t=1}^{T} r_{ijt}^{k} = M_{ij}.$$

One way to represent these requirements is to add a set of constraints which include the r_{ijt}^{k} and some zero-one variables. One possible set of such constraints is given below. In these constraints suppose that activity (i,j) must precede activity (j,p).

(III-C-6) $\quad \sum_{t=1}^{\tau} r_{ijt}^{k} - (M_{ij} - 1)\delta_{ij\tau} - M_{ij}\gamma_{ij\tau} \leq 0$

$$\text{for} \quad \tau = 1, 2, \ldots, T-1$$

(III-C-7) $\quad \sum_{t=1}^{\tau} r_{ijt}^{k} - \delta_{ij\tau} - M_{ij}\gamma_{ij\tau} \geq 0$

$$\text{for} \quad \tau = 1, 2, \ldots, T-1$$

(III-C-8) $\quad r_{j,p,\tau+1}^{q} - M_{jp}\gamma_{ij\tau} \leq 0$

$$\text{for} \quad \tau = 1, 2, \ldots, T-1$$

(III-C-9) $\quad \delta_{ij\tau} + \gamma_{ij\tau} \leq 1, \quad \tau = 1, 2, \ldots, T-1$

(III-C-10) $-\delta_{ij\tau} + \delta_{i,j,\tau+1} \geq 0, \quad \tau = 1, 2, \ldots, T-2$

(III-C-11) $\quad -\gamma_{ij\tau} + \gamma_{i,j,\tau+1} \geq 0, \quad \tau = 1, 2, \ldots, T-2$

$$\delta_{ij}, \quad \gamma_{ij} = 0 \text{ or } 1, \quad \forall (i,j), \tau.$$

A block of constraints such as (III-C-6)-(III-C-11) would be associated with each activity that is an immediate pre-decessor of another.

The inequalities (III-C-6) and (III-C-7) along with (III-C-9) state that the total allocation in $(0,\tau]$ is either less than M_{ij} and greater than zero, equal to zero, or is equal to M_{ij}. This when considered with (III-C-8) requires that activity (j,p) not be started until

$$\sum_{t=1}^{\tau} r_{ijt}^{k} = M_{ij}.$$

Inequality (III-C-10) requires an additional allocation in period $\tau+1$ if the activity was not completed in period τ. Inequality (III-C-11) is added to ensure that if activity (i,j) is completed in period τ then

$$\sum_{t=1}^{\tau} r_{ijt}^{k} = \sum_{t=1}^{\tau+1} r_{ijt}^{k} = \ldots = \sum_{t=1}^{T} r_{ijt}^{k} = M_{ij}.$$

Suppose these constraints are now represented in matrix form. The entire set of constraints can be represented by the matrix sum

$$D_1 x_1 + D_2 x_2 + \ldots + D_K x_K + Gy \geq d$$

52

where D_k is the matrix of coefficients of the allocations, G is the matrix of coefficients of the zero-one variables, and y is the column vector of zero-one variables. Additionally, d is the right hand side of the precedence constraints. The complete problem (M1) can then be written as

Minimize $c_1 x_1 + c_2 x_2 + \ldots + c_K x_K$

Subject to

$$A_1 x_1 \qquad\qquad\qquad = a_1$$

$$B_1 x_1 \qquad\qquad\qquad = b_1$$

(M1)

$$A_2 x_2 \qquad\qquad = a_2$$

$$B_2 x_2 \qquad\qquad = b_2$$

$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots$$

$$A_K x_K \;\; = a_K$$

$$B_K x_K \;\; = b_K$$

$$D_1 x_1 + D_2 x_2 + \ldots + D_K x_K + Gy \geq d$$

$$x_1, \; x_2, \; \ldots, \; x_K \geq 0$$

$$y_i \; \varepsilon \; y = 0 \text{ or } 1.$$

3. Partitioning Procedure

A solution procedure for (M1) can be developed by applying the Benders partitioning procedure [7]. This

procedure is described in Benders' paper as well as by Geoffrion and Marsten [33], Hu[39, p. 259], and Lasdon [51, p. 370].

For fixed y, (M1) can be partitioned yielding the linear programming problem

Minimize $\quad c_1 x_1 + \dots + c_K x_K$.

Subject to

(M2)

$$A_1 x_1 \qquad\qquad\qquad = a_1$$

$$B_1 x_1 \qquad\qquad\qquad = b_1$$

$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots$$

$$A_K x_K \;=\; a_K$$

$$B_K x_K \;=\; b_K$$

$$D_1 x_1 + \dots + D_K x_K \geq d - Gy$$

$$x_1, \; x_2, \; \dots, \; x_K \geq 0.$$

This is a large linear programming problem in block diagonal form. For fixed y this could, in principle, be solved using the Dantzig-Wolfe decomposition procedure [22], [23]. The dual of (M2) is

Maximize $\quad u_1 a_1 + v_1 b_1 + \ldots + u_K a_K + v_K b_K + w(d-Gy)$

Subject to

(M2D)
$$u_1 A_1 + v_1 B_1 \qquad\qquad\qquad\qquad + wD_1 \leq c_1$$

$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots$$

$$u_K A_K + v_K B_K + wD_K \leq c_K$$

$$u_1, \ldots, u_K, v_1, \ldots, v_K \text{ unrestricted}, w \geq 0.$$

For some extreme point, (u^p, v^p, w^p), feasible in (M2D), this
is

(M2D') Max $\quad u_1^p a_1 + v_1^p b_1 + \ldots + u_K^p a_K + v_K^p b_K + w^p(d-Gy)$.

Then, following Benders' procedure, this leads to

Minimize z

Subject to

(M1')

$$z \geq \underset{(u^p, v^p, w^p)}{\text{Max}} \{u_1^p a_1 + v_1^p b_1 + \ldots + w^p(d-Gy)\}$$

$$y_i \; \epsilon \; y = 0 \text{ or } 1$$

or equivalently

Minimize z

Subject to

(M1*)

$$z \geq u^p a + v^p b + \dots + w^p (d-Gy), \quad p = 1, 2, \dots, P$$

$$y_i \; \varepsilon \; y = 0 \text{ or } 1$$

where P is the number of extreme points of (M2D).

Now return to problem (M2D). If the partitioning procedure is again performed by fixing w then problem (S1) is obtained. It is

$$\text{Maximize} \quad u_1 a_1 + v_1 b_1 + \dots + u_K a_K + v_K b_K$$

Subject to

(S1)

$$u_1 A_1 + v_1 B_1 \qquad\qquad\qquad \leq c_1 - w D_1$$

$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots$$

$$u_K A_K + v_K B_K \leq c_K - w D_K$$

$$u_1, \dots, u_K, v_1, \dots, v_K \text{ unrestricted.}$$

The dual of (S1) is

56

Minimize $(c_1 - wD_1)r_1 + \ldots + (c_K - wD_K)r_K$

Subject to

(S1D)

$$A_1 r_1 \qquad\qquad\qquad = a_1$$

$$B_1 r_1 \qquad\qquad\qquad = b_1$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$A_K r_K \; = a_K$$

$$B_K r_K \; = b_K$$

$$r_1, \; r_2, \; \ldots, \; r_K \geq 0.$$

This is again K separate transportation problems. Once again, following Benders' procedure, for some extreme point $(r_1^q, \; \ldots, \; r_K^q)$ and combining (M2D) and (S1D) the problem becomes

Maximize Z

Subject to

(S1D')

$$Z \leq w(d - Gy) + \underset{(r_1^q, \; \ldots, \; r_K^q)}{\text{Min}} \{(c_1 - wD_1)r_1 + \ldots + (c_K - wD_K)r_K\}$$

$$w \geq 0.$$

This is a linear programming problem with variables w. The solution procedure can now be described with an algorithm.

57

4. Solution Procedure

The Benders partitioning procedure is applied twice to obtain the subproblems defined above. This double application of the procedure is now summarized by an algorithm.

Algorithm 1

STEP 1: Find an extreme point $(u_1, v_1, \ldots, u_K, v_K, w)$ of the feasible region of problem (M2D). Solve the K separate transportation problems (S1D) to get an initial $(r_1, \ldots, r_K)$. Go to STEP 2.

STEP 2: Solve the zero-one integer programming problem (M1*) for y, z using any zero-one code (e.g., Balas [1]). Go to STEP 3.

STEP 3: Using y from STEP 2 and current $(r_1, \ldots, r_K)$ solve the linear programming problem (S1D') for w. Go to STEP 4.

STEP 4: Using w from STEP 3 solve the linear programming problem (S1D) for a new $(r_1, \ldots, r_K)$. Denote it $(\hat{r}_1, \ldots, \hat{r}_K)$. If
$$Z - w(d-Gy) \leq (c_1 - wD_1)\hat{r}_1 + \ldots + (c_K - wD_K)\hat{r}_K$$
then go to STEP 5. Otherwise go to STEP 3 adding the additional constraint
$$Z \leq w(d-Gy) + (c_1 - wD_1)\hat{r}_1 + \ldots + (c_K - wD_K)\hat{r}_K.$$

STEP 5: Using w obtained in STEP 3, solve the linear programming problem (S1) for $(\hat{u}_1, \hat{v}_1, \ldots, \hat{u}_K, \hat{v}_K)$. If
$$z \geq \hat{u}_1 a_1 + \hat{v}_1 b_1 + \ldots + \hat{u}_K a_K + \hat{v}_K b_K + w(d-Gy)$$

58

then go to STEP 6. Otherwise go to STEP 2 adding
the constraint

$$z \geq \hat{u}_1 a_1 + \hat{v}_1 b_1 + \ldots + \hat{u}_K a_K + \hat{v}_K b_K + w(d-Gy)$$

to the existing set of constraints.

STEP 6: Using the y obtained in STEP 2, solve the linear
programming problem (M2). Let the optimal solution
to this be $(x_1^*, \ldots, x_K^*)$, then the optimal value of
the objective function is $c_1 x_1^* + \ldots + c_K x_K^*$. The
optimal allocations are given by $x_1^*, \ldots, x_K^*$.

It was proved in Reference 7 that the Benders par-
titioning procedure terminates after a finite number of iter-
ations with the optimal solution to a mixed integer linear
programming problem. The duration minimization algorithm
composed of two applications of this procedure then, is also
a finite method.

D. PATTON'S PROJECT SCHEDULING MODEL

 1. Model Description

In his doctoral dissertation [67], George Patton
proposed a mixed integer linear programming model for the
network job-shop scheduling problem. It is shown in the
next section that his model is a special case of the shipyard
scheduling model presented in this thesis. As mentioned in
Chapter II, a network job-shop scheduling problem is a resource
allocation problem in which all available resources of a par-
ticular type are employed on one activity at a time until it

is complete. Some additional notation is now necessary
before describing Patton's model.

Suppose that all of some resource k, k = 1, 2, ..., K
must be employed on one activity at a time. Each activity
that must be processed by this resource then has a demand for
a certain amount of processing time from that resource. In
addition, each resource is available for some period of time
during the project. Suppose d_{ij}^k denotes the total time
demand expressed for resource type k by activity (i,j) and
x_{ijt}^k denotes the amount of time spent on activity (i,j) by
resource type k during time period t. Then the amount of
time necessary to complete activity (i,j) is

(III-D-1) $$\sum_{t=1}^{T} x_{ijt}^k = d_{ij}^k ; \qquad \forall (i,j) \; \epsilon \; N$$
$$k = 1,2,...,K.$$

Also, if a_t^k is the total amount of time available in time
period t then

(III-D-2) $$\sum_{(i,j) \epsilon S_k} x_{ijt}^k \leq a_t^k ; \qquad k = 1,2,...,K$$
$$t = 1,2,...,T.$$

Note that (III-D-1) and (III-D-2) are also the constraints
for a transportation problem with inequalities as were
(III-C-1) and (III-C-2). The units of measurement differ
between these models however. Patton's model is expressed
in time units while the variables in the shipyard scheduling
model are resources per unit time. This permits expressing
a somewhat more general problem.

60

In order to minimize project duration, Patton intro-
duced a dummy activity (n,n+1). Associated with this dummy
activity is a unit time demand and availability of a dummy
resource. It was shown that if

(III-D-3) Minimize $\sum_{t=1}^{T} (t-1)x_{n,n+1,t}$

was used as the objective function with (III-D-1) and
(III-D-2) then the result was minimum project duration. The
relations

$$\sum_{t=1}^{T} x_{n,n+1,t} = 1$$

and

$$x_{n,n+1,t} \leq 1 ; \quad t = 1,2,\ldots,T$$

must, of course be added to the constraint set (III-D-1),
(III-D-2). This model accounted for activity precedence
relations by associating zero-one variables δ_{jt}, $t = 1,\ldots,T$
with each event j and adding the constraints

(III-D-4) $x_{jpt}^{q} \leq \delta_{jt}a_{t}^{q}$; $(j,p)\epsilon S_{q}$

$$t = 1, 2, \ldots, T$$

$$j = 1, 2, \ldots, n$$

(III-D-5) $\delta_{jt} \leq \dfrac{\sum_{k=1}^{K} \sum_{(i,j)\epsilon S_{k}} \sum_{s=1}^{t-1} x_{ijs}^{k}}{\sum_{k=1}^{K} \sum_{(i,j)\epsilon S_{k}} d_{ij}^{k}}$; $j=2,3,\ldots,n$

$$t=2,3,\ldots,T$$

$$\delta_{jt} = 0 \text{ or } 1.$$

The constraints (III-D-4) and (III-D-5) allow the interruption of work on an activity. For constraints of this nature to apply in this thesis, assumption 2 of Section B of this chapter must be relaxed.

   2.   Simplifying the Shipyard Scheduling Model

      If two additional assumptions are stated, the mixed integer linear program (III-C-1)-(III-C-3) can be stated in the form of Patton's model (III-D-1)-(III-D-5). Also if this model is to be valid, assumption 2 must be relaxed. This allows activities to be separated. The new assumptions that must be satisfied are concerned with the manner in which the shop personnel are employed. They are:

   3.   Each shop exhibits a constant resource profile.

   4.   All available personnel from a shop must work on a single activity at a time.

Assumption 3 states that the same number of workmen are available from a shop on any given point in time during the project. This assumption, along with assumption 4, amount to the fixing of a crew size from each shop working on an activity.

      Suppose the constant number of workers available from shop k is R^k. Also, it is known that each activity (i,j) in the project requires M_{ij} mandays to accomplish. If assumption 4 is satisfied and the time periods $t = 1,2,\ldots,T$ are single days, then for the shipyard problem, $a_t^k = 1$. That is, there is one day of resource type k available in any unit time period t. The time demand for activity (i,j) is the

62

number of days to meet the activity manday requirement M_{ij}.
This necessitates representing (III-C-2) as an inequality.
Then

$$d_{ij}^k = \left\{ \frac{M_{ij}}{R^k} \right\}$$

where {x} represents the smallest integer which is greater
than or equal to x. The mixed integer linear programming
model for shipyard project duration can then be expressed
directly as (III-D-1)-(III-D-5). The relaxation of assump-
tion 2 is necessary to permit the use of precedence con-
straints of the form (III-D-4), (III-D-5). Solution methods
for the problem when cast in this network job-shop scheduling
problem form appear in Chapter V.

3. Revised Precedence Relations

The constraints involving zero-one variables (III-D-4),
(III-D-5) can be adjoined to the shipyard project duration
model (III-C-1)-(III-C-3) without converting it to Patton's
model. Assumptions 3 and 4 need not be satisfied. Once
again, however, assumption 2 must be relaxed. The substitu-
tion of the revised constraints reduces the number of zero-
one variables and precedence constraints immensely. Instead
of associating two zero-one variables with each activity and
time period, one binary variable is associated with each
event and time period.

The revised constraints are, for event j and time period τ:

(III-D-6) $$r^q_{jp\tau} \leq \delta_{j\tau} R^q_\tau$$

(III-D-7) $$\sum_{t=1}^{\tau-1} \sum_{k=1}^{K} \sum_{(i,j)\epsilon S_k} r^k_{ijt} \geq [\sum_{(i,j)} M_{ij}] \delta_{j\tau}.$$

These constraints ensure that the logical requirement that each activity that must be completed prior to event j's attainment time is satisfied. Work on these activities may be interrupted and resumed.

The number of zero-one variables δ_{jt} and continuous variables r^k_{ijt} employed may be reduced by eliminating those which are unnecessary. For example, if two activities (i,j) and (j,p) are in series, activity (j,p) could never be processed in time period 1. Likewise (i,j) could never be processed in time period T. The observance of serial relationships such as these in a large network can reduce greatly the number of zero-one variables necessary to show precedence.

E. COST MINIMIZATION IN PROJECTS

1. Minimum Total Cost Formulation

In Chapter I it was stated that total project cost in the naval shipyard consists of the total direct labor cost (during normal working hours), overtime cost, and a penalty cost for exceeding a due date. In this section, the mixed integer linear programming model is extended to account for these costs.

Associated with each shop k and activity (i,j) is a cost c_{ij}^k. This is the cost in dollars for one manday of work, during the normal workday, by shop k personnel on activity (i,j). This cost is normally the same for each activity on which shop k is employed but occasionally may differ. Another labor cost, d_{ij}^k is the cost of an overtime manday employed on activity (i,j) by shop k. The third and final cost in the total project cost is a penalty cost C^* for exceeding some target date T^*. T^* is measured in days from project commencement as is project duration. There is an upper limit on the number of overtime mandays that can be expended. This bound is specified by shipyard management personnel and consists of a fixed percentage of total project mandays

$$\sum_{(i,j) \in N} M_{ij}.$$

The project duration can be represented in a manner similar to (III-D-3) if an artificial activity (n,n+1) is added. In order for (n,n+1) to be completed, one artificial resource unit must be expended during one time period. On any given day, only one unit of this fictitious resource is available. The project duration can then be expressed as

$$\text{Minimize} \quad \sum_{t=1}^{T} (t-1) r_{n,n+1,t}$$

The constraints

$$(\text{III-E-1}) \qquad \sum_{t=1}^{T} r_{n,n+1,t} = 1$$

and

(III-E-2) $\quad r_{n,n+1,t} \leq 1$; $\quad t = 1, 2, \ldots, T$

must be added to the constraint set (III-C-1), (III-C-2). The penalty cost can then be represented as

(III-E-3) $\quad C^* [\sum_{t=1}^{T} (t-1) r_{n,n+1,t} - T^*]$

where

$$C^*[x] = \begin{cases} C^*x & \text{for} \quad x \geq 0 \\ 0 & \text{for} \quad x < 0. \end{cases}$$

The artificial activity must, of course, be represented in the precedence constraints. The total cost of labor during the normal workday is represented by

(III-E-4) $\quad \sum_{k=1}^{K} \sum_{(i,j)\epsilon S_k} c_{ij}^k \sum_{t=1}^{T} r_{ijt}^k$

and the total overtime cost is

(III-E-5) $\quad \sum_{(i,j)\epsilon N} d_{ij}^k s_{ij}^k .$

In place of Equation (III-C-2), the constraints

(III-E-6) $\quad \sum_{t=1}^{T} r_{ijt}^k + s_{ij}^k = M_{ij}$; $\quad (i,j)\epsilon S_k$

$\qquad\qquad\qquad\qquad\qquad\qquad k = 1, 2, \ldots, K$

66

are added. If (III-E-6) is solved for

$$\sum_{t=1}^{T} r_{ijt}^{k}$$

and this result is substituted into (III-E-4), the normal workday cost coefficient becomes zero and the total overtime cost becomes

(III-E-7) $$\sum_{(i,j)\epsilon N} (d_{ij}^{k} - c_{ij}^{k}) s_{ij}^{k} \ .$$

The total overtime is constrained by an upper bound B. Thus, the constraint

(III-E-8) $$\sum_{(i,j)\epsilon N} s_{ij}^{k} \leq B$$

is added to the constraint set.

The resulting mixed integer linear programming problem is called the total cost problem and is labelled (TCP1). This problem is summarized below.

Minimize $\left\{ \sum_{(i,j)\epsilon N} (d_{ij}^k - c_{ij}^k) s_{ij}^k + C^* \left[\sum_{t=1}^{T} (t-1) r_{n,n+1,t} - T^* \right] \right\}$

Subject to

$$\sum_{t=1}^{T} r_{ijt}^k + s_{ij}^k = M_{ij} \; ; \; \forall (i,j) \epsilon N$$

$$\sum_{t=1}^{T} r_{n,n+1,t} = 1$$

(TCP1) $\qquad \sum_{(i,j)\epsilon S_k} r_{ijt}^k \le R_t^k \; ; \quad \begin{array}{l} k = 1, 2, \ldots, K \\ t = 1, 2, \ldots, T \end{array}$

$$\sum_{(i,j)\epsilon N} s_{ij}^k \le B$$

$$0 \le r_{n,n+1,t} \le 1 \; ; \quad t = 1, 2, \ldots, T$$

$$r_{jp\tau}^q \le \delta_{j\tau} R_\tau^q \; ; \quad \begin{array}{l} j = 1, 2, \ldots, n \\ \tau = 1, 2, \ldots, T \end{array}$$

$$\sum_{t=1}^{\tau-1} \sum_{k=1}^{K} \sum_{(i,j)\epsilon S_k} r_{ijt}^k \ge \left[\sum_{(i,j)} M_{ij} \right] \delta_{j\tau} \; ; \quad \begin{array}{l} j = 1, 2, \ldots, n \\ \tau = 1, 2, \ldots, T \end{array}$$

$$r_{ijt}^k \ge 0, \; s_{ij}^k \ge 0, \; \delta_{j\tau} = 0 \text{ or } 1.$$

Also it must be true that

$$C^*(x) = \begin{cases} C^* x & \text{for } x \ge 0 \\ 0 & \text{for } x < 0. \end{cases}$$

68

This could be represented in matrix form as was the minimum duration problem (M1). A similar solution procedure can then be applied to (TCP1).

2. Solution Procedure

The method for finding the minimum total cost of the project is an extension of Algorithm 1 in Section III-C-4. A provision must be made to ensure that the penalty cost term is nonzero when project duration exceeds the target and zero when the project completion time occurs before the target.

A procedure for incorporating this requirement is summarized in the following algorithm.

Algorithm 2

STEP 1: Using Algorithm 1 solve the minimum total cost problem (TCP1). Associated with solution is a project duration D. Go to STEP 2.

STEP 2: If (a) $D \geq T^*$ stop, the optimal solution has been obtained. Otherwise, go to STEP 3.

STEP 3: If (a) All $s_{ij}^k = 0$ stop, the optimal solution has been obtained. Otherwise, there exists an $s_{ij}^k > 0$, vary s_{ij}^k parametrically until $D = T^*$ then Stop with optimal solution.

F. SUMMARY

A method has been presented in this chapter that solves, in principle, the problem of minimizing project cost when resources are restricted and a fixed manday requirement must

69

be met. This chapter shows that a shipyard scheduling project can be represented by several transportation problems linked by precedence constraints. There is one transportation problem for each shop.

As a computational procedure the method is not presently feasible. One difficulty is in finding an initial basic feasible solution to problem (M2D) which is needed for the algorithm. Another problem is that a large zero-one programming problem must be solved in STEP 2 of Algorithm 1.

IV. NONLINEAR PROGRAMMING FORMULATION

A. INTRODUCTION

This chapter considers the problem of minimizing total project cost subject to several resource constraints. A known and fixed number of mandays must be allocated for completion of each activity in the project. A dynamic programming approach to a special case of this problem leads to a solution procedure for minimizing project duration. The Benders partitioning procedure [7] is then applied to the total cost problem and a solution method is obtained. Some characteristics of the multiple resource problem are explored.

Very few approaches to scheduling problems have been concerned with problems in which a specified number of resource-time units are needed to complete an activity. Bershchanskii [9] addressed two problems under this framework. The first was the minimization of project duration subject to a restriction on the total amount of various resources used. His other problem was to allocate the minimum quantity of resources to complete a project by a fixed due date. In both problems, a fixed number of resource-time units was specified for each activity. Mason and Moodie [58] developed a branch and bound for minimizing project cost. The costs involved were penalty costs for changing resource levels and exceeding a specified due date. The use of dynamic programming in solving scheduling problems of the type considered in this thesis was discussed by Petrovic [68]. A disadvantage of his approach was

71

the requirement that all event times be known and fixed. Razumikhin [72], [73], [74] attacked several problems in which a fixed number of mandays had to be used to complete each activity in a project. In two of these papers [72] and [74], a hydrostatic model was presented and solution methods using principles of fluid mechanics were given. The formulation of the total cost model in Section C of this chapter is an extension of the problem statement in the last two papers.

B. ASSUMPTIONS AND NOTATION

The following notation is used throughout this chapter:

(i,j) = The activity starting at event (node) i and terminating at event j.

m = The total number of activities in the network.

n = The total number of events in the network.

N = The collection of activities and events making up the project.

t_i = The time at which event i takes place.

c_{ij}^k = The normal cost per manday for shop k in the performance of activity (i,j).

d_{ij}^k = The overtime cost per manday for shop k in the performance of activity (i,j).

$r_{ij}^k(t)$ = The number of men from shop k employed on activity (i,j) at time t.

s_{ij}^k = The number of overtime mandays expended on activity (i,j) by shop k.

M_{ij} = The number of mandays required to accomplish activity (i,j).

R_i^k = The number of men from shop k available between events i and i+1.

K = The total number of shops.

B = The upper bound on total overtime mandays allowed, usually a fixed percentage of total project man-days, ΣM_{ij}.

T* = The due date for the project measured in days from project commencement time t_1.

C* = A penalty cost per day incurred when the project is completed after the due date.

The resources of type k are often referred to as men from shop k and the unit of time is considered to be a day. This is not critical to the development of the problem and is only done for convenience. Any resource type and resource-time unit might apply.

Four assumptions must be made in order that the development in this chapter be valid. The assumptions are:

1. The events of the project must be ordered 1, 2, ..., n such that $t_1 \leq t_2 \leq \ldots \leq t_n$ and if an activity (i,j) exists then $t_i < t_j$.

2. The project can be partitioned into subsets of activities P_1, P_2, ..., P_{n-1} where $N = \bigcup_{i=1}^{n-1} P_i$ and the subscripts correspond to the event times t_1, t_2, ..., t_{n-1}. The resource availabilities R_i^k, i = 1, 2, ..., n-1, are imposed over the time intervals $[t_i, t_{i+1})$, i = 1, 2, ..., n-1.

3. Only one resource type is required by each activity.

4. $r_{ij}^k(t)$ is a step function of t with the possible discontinuities occuring at the event times $t_i, t_{i+1}, \ldots, t_{j-1}$. That is, it is assumed that $r_{ij}^k(t_\alpha) = r_{ij}^k(t_{\alpha+1} - \varepsilon)$ for small $\varepsilon > 0$ and $\alpha = i, i+1, \ldots, j-1$.

To illustrate assumptions 1 and 2 consider the network in Figure 8. Only one resource type is needed for this network. R_1 is the amount of resource available between events i and i+1. In Figure 8 activity (1,3) can utilize the resources available in both $[t_1, t_2)$ and in $[t_2, t_3)$ where t_1, t_2, and t_3 are the times of occurrence of events 1, 2, and 3. Assumptions 1 and 2 are quite severe for many projects. Some possible methods for making these two assumptions more palatable are disucssed in Section F.

C. MODEL DEVELOPMENT

The development can now proceed. The objective is to find an allocation of men from the various shops that will minimize the total cost of completing the project. Total project cost is considered to be the sum of direct labor costs, overtime costs, and the penalty cost incurred when the due date is exceeded. The specified number of mandays required for completion of each activity must be utilized. The sum of normal and overtime mandays expended must equal this number. Constraints are imposed on the number of normal mandays that can be expended in each inter-event interval. Finally there is an upper bound on the total number of overtime mandays which can be expended in completing the project.
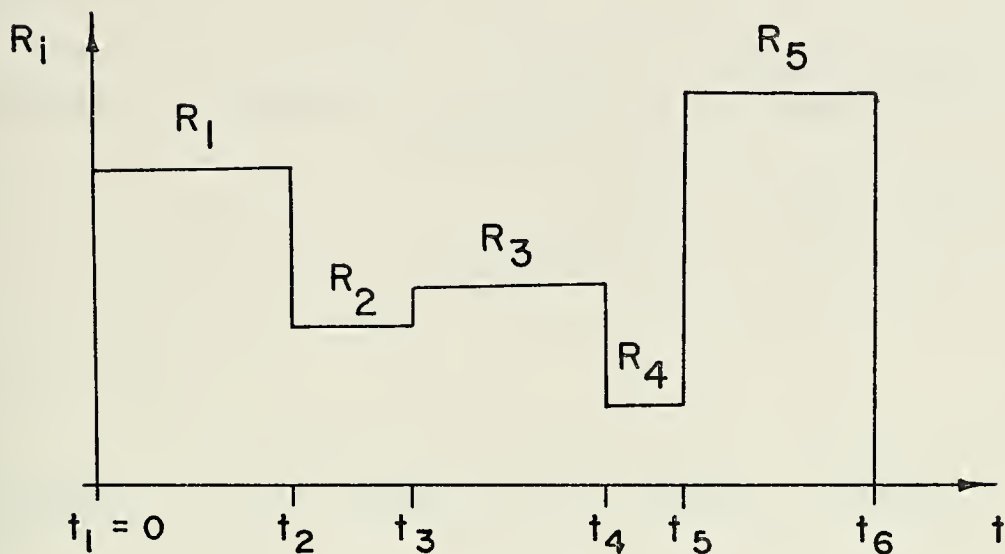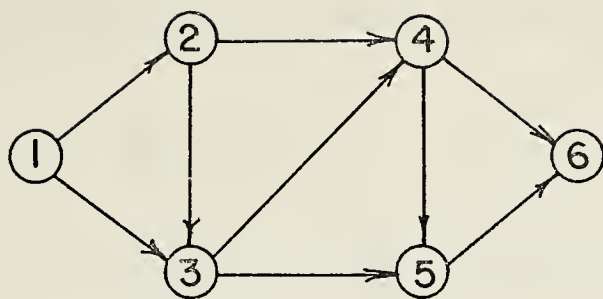
Figure 8

The total number of mandays expended on activity (i,j) can be represented by the integral

$$\int_{t_1}^{t_n} r_{ij}^k(t)\ dt.$$

Outside the interval $[t_i,\ t_j)$ the resource allocation on activity (i,j) is zero. That is

$$r_{ij}^k(t) = 0 \text{ for } t < t_i \text{ and } t \geq t_j.$$

75

Employment during the normal workday is then represented by

$$\int_{t_i}^{t_j} r_{ij}^k(t)\, dt.$$

The following problem then acts as a model for this situation:

(IV-C-1)   Minimize $\left[\displaystyle\sum_{(i,j)\varepsilon N} \left[c_{ij}^k \int_{t_i}^{t_j} r_{ij}^k(t)\, dt\right]\right.$

$$\left. + \sum_{(i,j)\varepsilon N} d_{ij}^k s_{ij}^k + C^*(t_n - T^*)\right]$$

Subject to

(IV-C-2) $\displaystyle\int_{t_i}^{t_j} r_{ij}^k(t)\, dt + s_{ij}^k = M_{ij}\ ;\quad \forall (i,j)\varepsilon N$

(IV-C-3) $\displaystyle\sum_{(i,j)\varepsilon P_q} r_{ij}^k(t) \le R_q^k\ ;$ $k = 1, 2, \ldots, K$
                                                 $q = 1, 2, \ldots, n-1$
                                                 $t\ \varepsilon\ [t_q, t_{q+1})$

(IV-C-4) $\displaystyle\sum_{(i,j)\varepsilon N} s_{ij}^k \le B$

(IV-C-5) $s_{ij}^k \ge 0,\ t_i \ge 0,\ t_1 = 0,\ r_{ij}^k(t) \begin{cases} > 0 \text{ for } t_i \le t < t_j \\ \equiv 0 \text{ for } t < t_i \text{ or} \\ \qquad\qquad t \ge t_j \end{cases}$

and

(IV-C-6) $C^*(x) = \begin{cases} C^*x \text{ for } x \ge 0 \\ 0 \ \text{ for } x < 0. \end{cases}$

76

The objective function (IV-C-1) then represents the total cost. Equation (IV-C-2) and the inequalities (IV-C-3), (IV-C-4) represent the constraints specified in the problem statement. Equations (IV-C-6) ensure that the penalty cost is only incurred when the due date is exceeded. If the resource involved is men then $r_{ij}^k(t)$ must also be an integer.

If Assumption 4 is now adhered to, each of the integrals representing normal mandays expended become summations of the form

$$\int_{t_i}^{t_j} r_{ij}^k(t) \, dt = \sum_{\alpha=i}^{j-1} r_{ij}^k(t_\alpha)[t_{\alpha+1} - t_\alpha]$$

where $t_{i+1}, \ldots, t_{j-1}$ are the event times that occur while activity (i,j) is in progress. (IV-C-1)-(IV-C-3) then become

(IV-C-7)   Minimize $\{ \sum_{(i,j)\epsilon N} c_{ij}^k [\sum_{\alpha=i}^{j-1} r_{ij}^k(t_\alpha)[t_{\alpha+1} - t_\alpha]]$

$+ \sum_{(i,j)\epsilon N} d_{ij}^k s_{ij}^k + C^*(t_n - T^*)\}$

Subject to

(IV-C-8) $\sum_{\alpha=i}^{j-1} r_{ij}^k(t_\alpha)[t_{\alpha+1} - t_\alpha] + s_{ij}^k = M_{ij}; \quad \forall (i,j)\epsilon N$

(IV-C-9) $\sum_{(i,j)\epsilon P_q} r_{ij}^k(t_q) \leq R_q^k ; \quad \begin{matrix} k = 1, 2, \ldots, K \\ q = 1, 2, \ldots, n-1. \end{matrix}$

77

(IV-D-1) Minimize $\left\{ \displaystyle\sum_{(i,j)\epsilon N} c_{ij}^k \left[\displaystyle\sum_{\alpha=i}^{j-1} r_{ij}(t_\alpha)[t_{\alpha+1} - t_\alpha] \right] \right.$

$$\left. + C^* t_n \right\}$$

Subject to

(IV-D-2) $\displaystyle\sum_{\alpha=i}^{j-1} r_{ij}(t_\alpha)[t_{\alpha+1} - t_\alpha] = M_{ij} \; ; \quad \forall (i,j) \epsilon N$

(IV-D-3) $\displaystyle\sum_{(i,j)\epsilon P_q} r_{ij}(t_q) \leq R_q \; ; \quad q = 1,2,\ldots,n-1$

(IV-D-4) $r_{ij}(t_\alpha) \geq 0 \; ; \quad \alpha = i,i+1,\ldots,j-1$

$t_1 = 0, \; t_i \geq 0 \; ; \quad i = 2,3,\ldots,n.$

This problem can now be represented as a complex feed-forward loop system and can be described by a block diagram [64, p. 204]. Figure 10 is an example of this for the network of Figure 9. Note that in Figure 9 the events can be ordered so that $t_1 < t_2 < t_3 < t_4$.
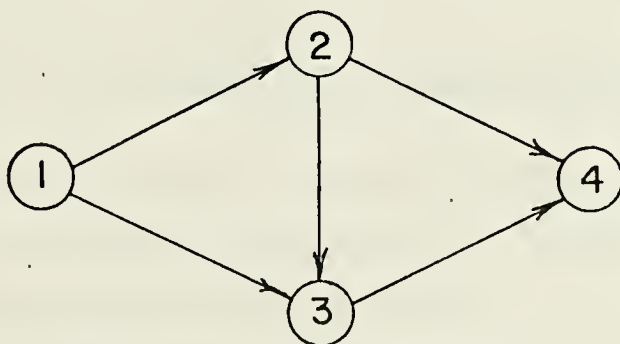
Figure 9

The relations (IV-C-4)-(IV-C-6) remain the same. The problem as formulated by (IV-C-5)-(IV-C-9) is a nonlinear programming problem which is very difficult to solve. This total cost formulation is an extension of the model proposed by Razumikhin [72], [74]. In [72] the goal was to level resources. This was done by minimizing the square deviation of resource expenditures from a constant. It was shown in that paper that this was equivalent to minimizing the maximum resource allocation. The objective of [74] was to minimize project duration. Principles of fluid mechanics were utilized to provide a solution procedure in both papers. Razumikhin's mechanical model is described below in Section IV-G. The normal resource constraints of the total cost problem (IV-C-8)-(IV-C-9) are similar to resource constraints in [72] and [74] with no overtime included.

This chapter first considers the case wehre overtime is not used (B = 0). It is then shown that solving this form of the problem becomes one of minimizing project duration. The results are then extended to the total cost problem.

D. DYNAMIC PROGRAMMING APPROACH

1. <u>Duration Minimization for a Single Constrained Resource</u>

In this section only a single resource is required by all of the activities in the project. In addition, over-time is temporarily omitted from the model. The reduced total cost model then becomes

figure 10
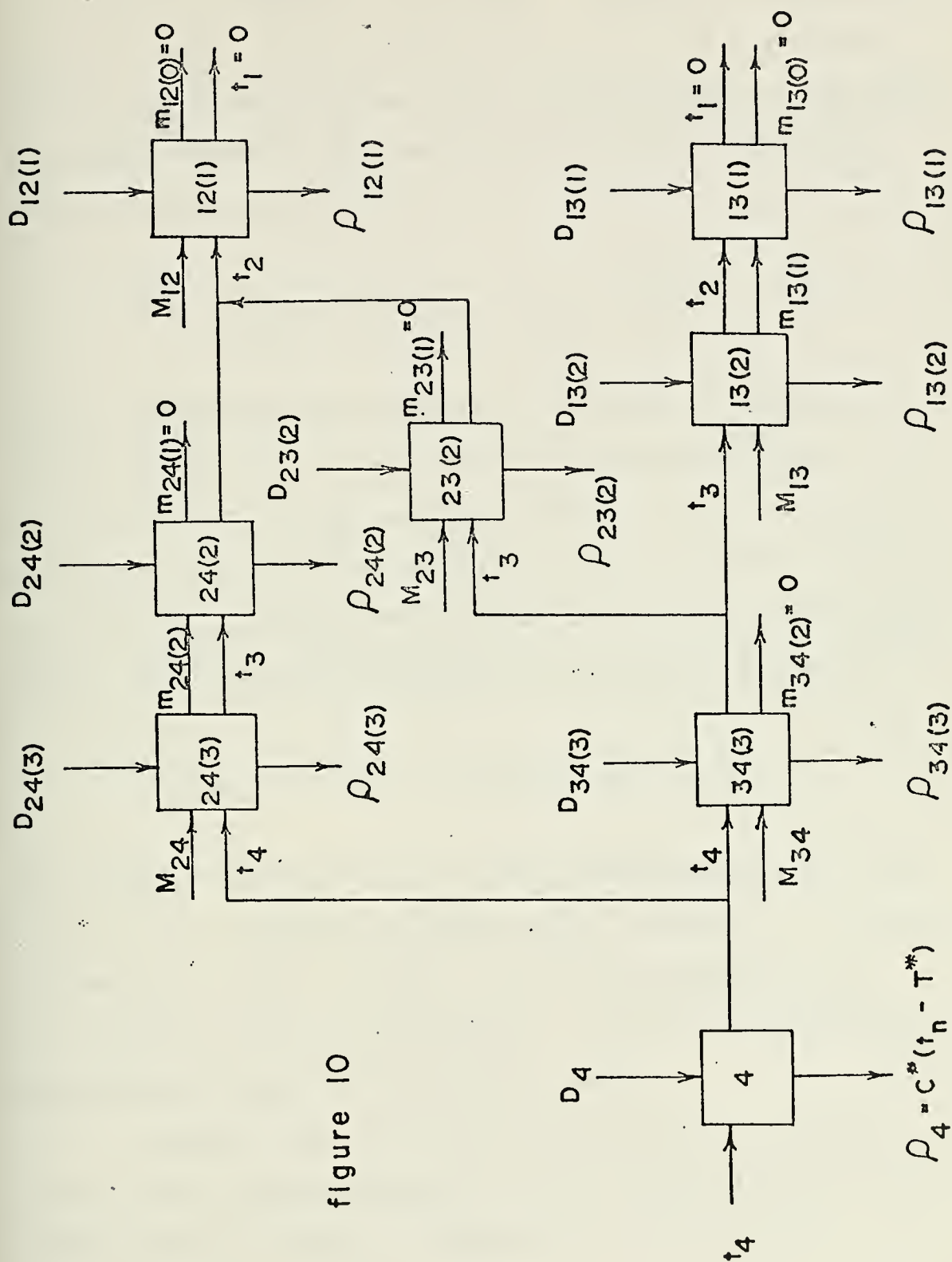
In the dynamic programming formulation each stage represents a portion of an activity. Stage ij(k) stands for activity (i,j) at the breakpoint t_k ε [t_i, t_j). The decision $D_{ij(k)}$ represents the number of men assigned to activity (i,j) over the time interval [t_k, t_{k+1}). The individual stage return is $\rho_{ij(k)}$ and

$$\rho_{ij(k)} = c_{ij} D_{ij(k)} [t_{k+1} - t_k].$$

Stage n represents the decision to increase or decrease project duration t_n. For each of the stages there are two state variables, $m_{ij(k)}$ and t_k. The state variable $m_{ij(k)}$ represents the number of mandays available for input to stage ij(k) while t_k is the time of event k output from all stages ij(k). The stage transformation for $m_{ij(k)}$ is given by

$$m_{ij(k-1)} = m_{ij(k)} - D_{ij(k)} [t_{k+1} - t_k].$$

The stage transformation for time is described later. The serial system composed of stages ij(i), ij(i+1), ..., ij(j-1) represents a single activity. The state variable $m_{ij(k)}$ only appears in this serial system. On the other hand, time connects all stages of the complex system.

Consider now an arbitrary activity (i,j) and the serial system associated with it. The block diagram for this serial system is shown in Figure 11.

81
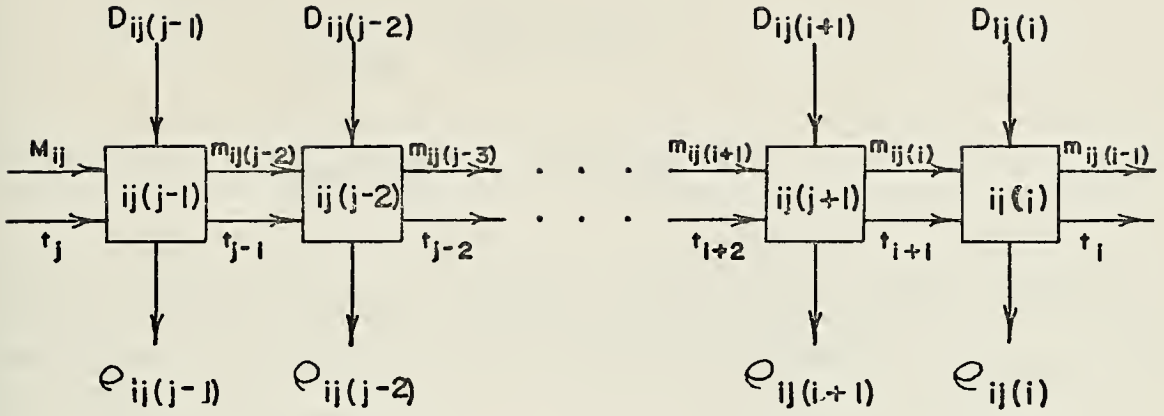
Figure 11

Beginning with stage $ij(i)$ the minimum $ij(i)$, $ij(i+1)$, ..., $ij(ij(j-1)$ stage returns can be developed. Denote the minimum $ij(k)$ stage return by $f_{ij(k)}(m_{ij(k)}, t_k, t_{k+1})$. Then for stage $ij(i)$

(IV-D-5) $f_{ij(i)}(m_{ij(i)}, t_i, t_{i+1}) =$

$$\underset{D_{ij(i)}}{\text{Min}} \quad \rho_{ij(i)}(m_{ij(i)}, t_i, t_{i+1}, D_{ij(i)})$$

Subject to

$$m_{ij(i-1)} = m_{ij(i)} - D_{ij(i)}[t_{i+1} - t_i]$$

and

$$\sum_{(k,l)\epsilon P_i} D_{kl(i)} \leq R_i$$

$$D_{kl(i)} \geq 0; \quad (k,l)\epsilon P_i.$$

Since (IV-D-2) requires equality, the total number of man-days remaining at stage ij(i) must be expended, $m_{ij(i-1)} = 0$ and (IV-D-5) becomes

$$f_{ij(i)}(m_{ij(i)},\, t_i,\, t_{i+1}) = \underset{D_{ij(i)}}{\text{Min}}\ c_{ij}D_{ij(i)}[t_{i+1} - t_i]$$

(IV-D-6)   Subject to

$$D_{ij(i)} = \begin{cases} \dfrac{m_{ij(i)}}{t_{i+1} - t_i} ; & t_{i+1} > t_i \\[2em] 0 ; & \text{otherwise} \end{cases}$$

and

$$\sum_{(k,l)\epsilon P_i} D_{kl(i)} \leq R_i.$$

This is simply

(IV-D-7) $$f_{ij(i)}(m_{ij(i)},\, t_i,\, t_{i+1}) = c_{ij}m_{ij(i)}$$

with

$$\frac{m_{ij(i)}}{t_{i+1} - t_i} + \underset{(k,l)\neq(i,j)}{\Sigma(k,l)\epsilon P_i} D_{kl(i)} \leq R_i.$$

Proceeding to stage $ij(i+1)$

$$f_{ij(i+1)}(m_{ij(i+1)}, t_{i+1}, t_{i+2})$$

$$= \underset{D_{ij(i+1)}}{Min} [f_{ij(i)}(m_{ij(i+1)}, t_{i+1}, t_{i+2})$$

$$+ \rho_{ij(i+1)}(m_{ij(i+1)}, t_{i+1}, t_{i+2}, D_{ij(i+1)})]$$

Subject to

$$m_{ij(i)} = m_{ij(i+1)} - D_{ij(i+1)}[t_{i+2} - t_{i+1}]$$

and

$$\underset{(k,l)\epsilon P_{i+1}}{\Sigma} D_{kl(i+1)} \leq R_{i+1}.$$

Following the development of (IV-D-7) this becomes

(IV-D-8) $f_{ij(i+1)}(m_{ij(i+1)}, t_{i+1}, t_{i+2}) = c_{ij}m_{ij(i+1)}$

with

$$\frac{m_{ij(i+1)} - m_{ij(i)}}{t_{i+2} - t_{i+1}} + \underset{(k,l)\neq(i,j)}{\Sigma(k,l)\epsilon P_{i+1}} D_{kl(i+1)} \leq R_{i+1}.$$

84

It can be shown by induction that each succeeding stage exhibits this same performance and we have for the final stage

$$(IV-D-9) \qquad f_{ij(j-1)}(M_{ij}, t_{j-1}, t_j) = c_{ij}M_{ij}$$

with

$$\frac{M_{ij} - m_{ij(j-2)}}{t_j - t_{j-1}} + \Sigma_{\substack{(k,l) \ P_{j-1} \\ (k,l)\neq(i,j)}} D_{kl(j-1)} \leq R_{j-1}.$$

The choice of activity (i,j) was arbitrary and each activity in the network exhibits the same behavior. The optimum stage return for each stage is independent of the time input to and output from the stage. The stage decisions can be summarized as follows:

(a) If an activity is commenced at event i then the decision is

$$D_{ij(i)} = \frac{m_{ij(i)}}{t_{i+1} - t_i} .$$

(b) If event k is an intermediate event for activity (i,j) then

$$D_{ij(k)} = \frac{m_{ij(k)} - m_{ij(k-1)}}{t_{k+1} - t_k} .$$

85

(c) If an activity terminates at event j then

$$D_{ij(j-1)} = \frac{M_{ij} - m_{ij(j-2)}}{t_j - t_{j-1}} .$$

(d) If $t_k = t_{k+1}$ then $D_{ij(k)} = 0.$

Each resource constraint

$$\sum_{(i,j)\epsilon P_q} D_{ij(q)} \leq R_q ; \quad q = 1, 2, \ldots, n-1$$

then takes the form

$$\sum_{(q,j)} \frac{m_{qj(q)}}{t_{q+1} - t_q} + \sum_{\substack{(i,j) \\ i<q<j}} \frac{m_{ij(q)} - m_{ij(q-1)}}{t_{q+1} - t_q}$$

$$+ \sum_{(i,q+1)} \frac{M_{i,q+1} - m_{i,q+1(q)}}{t_{q+1} - t_q} \leq R_q; \quad q = 1, 2, \ldots, n-1.$$

This leads to the recursion

$$t_{q+1} \geq t_q + \frac{1}{R_q} \left[\sum_{(q,j)} m_{qj(q)} + \sum_{\substack{(i,j) \\ i<q<j}} (m_{ij(q)} - m_{ij(q-1)}) \right.$$

(IV-D-10)
$$\left. + \sum_{(i,q+1)} (M_{i,q+1} - m_{i,q+1(q)}) \right]$$

for $q = 1, 2, \ldots, n-1$ and $R_q > 0.$ Then (IV-D-10) is the
stage transformation for the event times.

This can now be simplified. Let

(IV-D-11) $x_{ij(k)} = m_{ij(k)} - m_{ij(k-1)}$; $k = i, i+1, \ldots, j-2$

and

(IV-D-12) $x_{ij(j-1)} = M_{ij} - m_{ij(k-2)}$.

Substituting (IV-D-11) and (IV-D-12) into (IV-D-10)

$$t_{q+1} \geq t_q + \frac{1}{R_q} [\sum_{(q,j)} x_{qj(q)} + \sum_{\substack{(i,j) \\ i<q<j}} x_{ij(q)} + \sum_{(i,q+1)} x_{i,q+1(q)}]$$

for $q = 1, 2, \ldots, n-1$ and $R_q > 0$. Or, this can be written

(IV-D-13) $$t_{q+1} \geq t_q + \frac{1}{R_q} [\sum_{\substack{(i,j) \\ i<q<j}} x_{ij(q)}].$$

This leads to a useful expression for project duration.

<u>Theorem</u>:   The project duration t_n satisfies the inequality

(IV-D-14) $$t_n \geq \sum_{(i,j)} \sum_{\alpha=i}^{j-1} \frac{1}{R_\alpha} x_{ij(\alpha)}.$$

Proof: Put $t_1 = 0$. From (IV-D-13)

$$t_2 \geq \frac{1}{R_1} \sum_{(1,j)} x_{1j(1)}.$$

Now assume that

$$t_{n-1} \geq \sum_{q=1}^{n-2} \frac{1}{R_q} \sum_{\substack{(i,j) \\ i<q<j}} x_{ij(q)}.$$

Then from (IV-D-13)

$$t_n \geq t_{n-1} + \frac{1}{R_{n-1}} \sum_{\substack{(i,j) \\ i<q<j}} x_{ij(n-1)}$$

and after rearranging terms

$$t_n \geq \sum_{(i,j)} \sum_{\alpha=i}^{j-1} \frac{1}{R_\alpha} x_{ij(\alpha)}$$

and the theorem is proved.

The total project cost is given by

$$\sum_{(i,j) \in N} c_{ij} M_{ij} + C^*(t_n - T^*)$$

and cost is minimized when project duration is minimized.

2. <u>Linear Programming Solution</u>

The theorem of Section D-1 gives an expression that project duration must satisfy. In addition to (IV-D-14), the total number of mandays consumed by an activity (i,j) must be M_{ij}. The minimum project duration subject to the resource restrictions can then be found by solving the linear programming problem

88

(IV-D-15)   Minimize $\displaystyle \sum_{(i,j)} \sum_{\alpha=i}^{j-1} \frac{1}{R_\alpha} x_{ij(\alpha)}$

Subject to

(IV-D-16) $\displaystyle \sum_{\alpha=i}^{j-1} x_{ij(\alpha)} = M_{ij} \; ; \quad \forall (i,j) \epsilon N$

$$x_{ij(\alpha)} \geq 0.$$

This is simply m independent linear programming problems each with a single constraint. The solution, therefore, is to put

(IV-D-17) $x_{ij(\hat\alpha)} = M_{ij}$

for

$$\frac{1}{R_{\hat\alpha}} = \underset{\alpha=i,\ldots,j-1}{\text{Min}} \{\frac{1}{R_\alpha}\}$$

and

(IV-D-18) $x_{ij(\alpha)} = 0$ otherwise.

Each event time t_q, $q = 1, 2, \ldots, n$ can then be determined using (IV-D-13). The resource allocations are then given by

(IV-D-19) $\displaystyle r_{ij}(t_\alpha) = \frac{x_{ij(\alpha)}}{t_{\alpha+1} - t_\alpha} , \quad \alpha = i, i+1, \ldots, j-1.$

89

Up to this point the only requirement placed on $r_{ij}(t_\alpha)$ has been nonnegativity. If the resource to be allocated is manpower, then $r_{ij}(t_\alpha)$ must also be integer. The solution to the linear programming problem (IV-D-15), (IV-D-16) guarantees that $x_{ij(\alpha)}$, $\alpha = i, i+1, \ldots, j-1$ are integer valued but $r_{ij}(t_\alpha)$ found from (IV-D-19) will not, in general, be integral. Therefore, under the present assumptions, the procedure is not valid for manpower allocation and some revised procedure must be used.

Suppose $r_{ij}(t_\alpha)$ is thought of as the time average manpower allocated over the time interval $[t_\alpha, t_{\alpha+1})$. Then define another variable $r^*_{ij}(t_{\delta_1})$ to be the actual manpower allocation over the time interval $[t_{\delta_1}, t_{\delta_2}) \subseteq [t_\alpha, t_{\alpha+1})$ and require that it be an integer. For the activity (i,j) to be completed it is necessary that

$$r^*_{ij}(t_{\delta_1})[t_{\delta_2} - t_{\delta_1}] = M_{ij}.$$

An expression for $r_{ij}(t_\alpha)$ can now be given as

(IV-D-20) $$r_{ij}(t_\alpha) = \frac{1}{t_{\alpha+1} - t_\alpha} \int_{t_\alpha}^{t_{\alpha+1}} r^*_{ij}(t) \, dt.$$

One possible value for $r^*_{ij}(t_{\delta_1})$ might be to set

$$r^*_{ij}(t_{\delta_1}) = R_\alpha \; ; \; \forall (i,j) \epsilon N.$$

That is, use all available resources in the performance of
activity (i,j) until it is completed. Then, if time $t_{\alpha+1}$
has not been reached, employ R_α men on some other activity
which is not a predecessor of activity (i,j). This is
continued until all activities have been scheduled.

It is necessary to show that this revised procedure
does not change the cost or the duration of the project.
Putting

$$r^*_{ij}(t_{\delta_1}) = R_\alpha$$

and carrying out the integration (IV-D-20)

$$r_{ij}(t_\alpha) = \frac{R_\alpha[t_{\delta_2} - t_{\delta_1}]}{t_{\alpha+1} - t_\alpha} .$$

Since M_{ij} mandays are necessary for completion of (i,j)

$$r_{ij}(t_\alpha)[t_{\alpha+1} - t_\alpha] = R_\alpha(t_{\delta_2} - t_{\delta_1}) = M_{ij}$$

and

$$t_{\delta_2} = t_{\delta_1} + \frac{M_{ij}}{R_\alpha}$$

for

$$t_\alpha \leq t_{\delta_1} \leq t_{\delta_2} \leq t_{\alpha+1}.$$

This procedure is carried out for every (i,j) in P_α until each is completed. For the first (i,j) under consideration in P_α put $\delta_1 = \alpha$. Commence the next activity at t_{δ_2} and continue this way until all activities are completed. Then

$$t_{\alpha+1} = t_\alpha + \sum_{(i,j)\epsilon P_\alpha} \frac{M_{ij}}{R_\alpha}$$

which is the same as the result given by the solution to the linear programming problem (IV-D-15), (IV-D-16). This method insures that an integer number of men is employed on each activity. An example of this appears in Section E-2.

E. COST MINIMIZATION

1. Cost Minimization for a Single Constrained Resource

The results of the previous sections are now extended to the problem of minimizing the total cost in a project subject to constraints on a single resource over time. The total cost problem (TCP2) for a single resource can be stated as

(IV-E-1)   Minimize $\{ \sum_{(i,j)\epsilon N} c_{ij} [\sum_{\alpha=i}^{j-1} r_{ij}(t_\alpha)[t_{\alpha+1} - t_\alpha]]$

$+ \sum_{(i,j)\epsilon N} d_{ij}s_{ij} + C^*(t_n - T^*) \}$

Subject to

(IV-E-2) $\sum_{\alpha=1}^{j-1} r_{ij}(t_\alpha)[t_{\alpha+1} - t_\alpha] + s_{ij} = M_{ij} \; ; \; \forall(i,j)\epsilon N$

(IV-E-3) $\qquad \underset{(i,j)\epsilon P_q}{\Sigma} r_{ij}(t_q) \leq R_q$; $q = 1, 2, \ldots, n-1$

(IV-D-4) $\qquad \underset{(i,j)\epsilon N}{\Sigma} s_{ij} \leq B$

(IV-E-5) $\qquad s_{ij} \geq 0,\ t_i \geq 0,\ t_1 = 0,\ r_{ij}(t) \begin{cases} \geq 0 \text{ for } t_i \leq t < t_j \\ \equiv 0 \text{ for } t < t_i \text{ or } \\ \qquad\qquad t \geq t_j \end{cases}$

and

(IV-E-6) $\qquad C^*(x) = \begin{cases} C^*x \text{ for } x \geq 0 \\ 0 \ \text{ for } x < 0. \end{cases}$

Note that from (IV-E-2)

(IV-E-7) $\qquad \overset{j-1}{\underset{\alpha=i}{\Sigma}} r_{ij}(t_\alpha)[t_{\alpha+1} - t_\alpha] = M_{ij} - s_{ij}.$

Then (IV-E-7) can be used to replace the first term of (IV-E-1). The objective function then becomes

Minimize $\qquad \underset{(i,j)\epsilon N}{\Sigma} c_{ij}[M_{ij} - s_{ij}] + \underset{(i,j)\epsilon N}{\Sigma} d_{ij}s_{ij} + C^*(t_n - T^*)$

then, collecting terms and ignoring constant terms, this becomes

(IV-E-8) $\qquad$ Minimize $\underset{(i,j)\epsilon N}{\Sigma} [d_{ij} - c_{ij}]s_{ij} + C^*(t_n - T^*)$.

Suppose now that all s_{ij} are fixed at some feasible value, say s_{ij}^*, $\forall (i,j) \epsilon N$. That is,

$$\sum_{(i,j) \epsilon N} s_{ij}^* \leq B$$

and

$$0 \leq s_{ij}^* \leq M_{ij} \; ; \quad \forall (i,j) \epsilon N.$$

Call (IV-E-8) and (IV-E-2)-(IV-E-6) the total cost problem and label it (TCP2). The Benders partitioning procedure [7] can then be applied to (TCP2) after transposing all the fixed terms involving overtime to the right hand side. For $T^* < t_n$, the problem becomes

(IV-E-9) $\qquad$ Minimize $C^* t_n$

Subject to

(IV-E-10) $\quad \displaystyle\sum_{\alpha=i}^{j-1} r_{ij}(t_\alpha)[t_{\alpha+1} - t_\alpha] = (M_{ij} - s_{ij}^*) \; ; \quad \forall (i,j) \epsilon N$

(IV-E-11) $\quad \displaystyle\sum_{(i,j) \epsilon P_q} r_{ij}(t_q) \leq R_q \; ; \quad q = 1, 2, \ldots, n-1.$

From Section IV-D-2 the problem (IV-E-9)-(IV-E-11) is equivalent to

$$\text{Minimize} \quad C^*[\sum_{(i,j)\epsilon N} \sum_{\alpha=i}^{j-1} \frac{1}{R_\alpha} x_{ij}(\alpha)]$$

Subject to

$$(ELP1) \quad \sum_{\alpha=i}^{j-1} x_{ij}(\alpha) = (M_{ij} - s^*_{ij}) \; ; \; \forall(i,j)\epsilon N$$

$$x_{ij}(\alpha) \geq 0.$$

The equivalent problem (ELP1) is a linear programming problem with solution given by (IV-D-17), (IV-D-18) if M_{ij} is replaced by $(M_{ij} - s^*_{ij})$ for every activity. Since (ELP1) is a linear programming problem, it has a dual which is

$$(IV-E-12) \quad \text{Maximize} \quad \sum_{(i,j)\epsilon N} (M_{ij} - s^*_{ij})u_{ij}$$

Subject to

$$(IV-E-13) \quad u_{ij} \leq C^* \frac{1}{R_\alpha} \; ; \quad \alpha = i, i+1, \ldots, j-1$$

$$u_{ij} \text{ unrestricted.}$$

The dual constraint (IV-E-13) is equivalent to

$$(IV-E-14) \quad u_{ij} \leq \frac{C^*}{\underset{\alpha=i,\ldots,j-1}{\text{Max}} R_\alpha}.$$

As explained in [7] the feasible region of (IV-E-12),
(IV-E-14) does not depend on s_{ij} and an optimal solution to
this system always occurs at an extreme point of the feasible
region. Additionally, since it is required that $0 \leq s_{ij} \leq M_{ij}$,
$(i,j)\epsilon N$ then $(M_{ij} - s_{ij}^*) \geq 0$, $(i,j)\epsilon N$ and the optimal
solution is

(IV-E-15)
$$u_{ij} = \frac{C^*}{\underset{\alpha=i,\ldots,j-1}{\text{Max}} R_\alpha} .$$

We then have

$$\text{Max} \underset{(i,j)\epsilon N}{\Sigma} [M_{ij} - s_{ij}^*]u_{ij} = \text{Min } C^*t_n.$$

Following Benders' procedure combine (IV-E-12), (IV-E-14)
(with u_{ij} given by (IV-E-15)) with the original problem
(TCP2) to obtain

Minimize Z

Subject to

$$Z \geq \{ \underset{(i,j)\epsilon N}{\Sigma} [d_{ij} - c_{ij}]s_{ij}$$

$$+ \text{Max } [\underset{(i,j)\epsilon N}{\Sigma} [M_{ij} - s_{ij}] \frac{C^*}{\underset{\alpha=i,\ldots,j-1}{\text{Max}} R_\alpha}]\}$$

$$\underset{(i,j)\epsilon N}{\Sigma} s_{ij} \leq B$$

$$0 \leq s_{ij} \leq M_{ij} ; \forall(i,j)\epsilon N.$$

This is equivalent to the linear programming problem

$$\text{Minimize} \quad \sum_{(i,j)\epsilon N} h_{ij} s_{ij}$$

Subject to

(ELP2)
$$\sum_{(i,j)\epsilon N} s_{ij} \leq B$$

$$0 \leq s_{ij} \leq M_{ij} ; \quad (i,j)\epsilon N$$

where

$$h_{ij} = [d_{ij} - c_{ij} - \frac{C^*}{\underset{\alpha=i,\ldots,j-1}{\text{Max}} R_\alpha}] ; \quad \forall (i,j)\epsilon N.$$

The overtime allocation is easily found by solving (ELP2). It is then a simple matter to find the normal resource allocations $r_{ij}(t_\alpha)$ by solving (ELP1). The procedure described above is summarized by an algorithm and an example is explained in the next section.

2. An Algorithm and an Example

Before the procedure is summarized, it is necessary to prepare for one contingency. If the solution to the total cost problem yields a project duration less than the due date, the relation (IV-E-6) has been violated. It is then necessary to reallocate overtime until the project duration and the due date become coincident. This is done by decreasing the most expensive overtime until this condition is met.

Algorithm 3

STEP 1: Put $s_{ij} = 0$; $(i,j) \epsilon N$. Go to STEP 2.

STEP 2: Put $x_{ij(\hat{a})} = (M_{ij} - s_{ij})$ for $1/R_{\hat{a}} = \underset{\alpha=i,\ldots,j-1}{\text{Min}} 1/R_{\alpha}$;

$x_{ij(\alpha)} = 0$, otherwise for $(i,j) \epsilon N$. Find the project duration

$$t_n = \underset{(i,j) \epsilon N}{\Sigma} \underset{\alpha=i}{\overset{j-1}{\Sigma}} (1/R_{\alpha}) x_{ij(\alpha)}.$$

(a) If $t_n = T^*$ or $t_n < T^*$ and all $s_{ij} = 0$ then terminate the algorithm. The present value of $x_{ij(\alpha)}$, $\forall (i,j) \epsilon N$, $\alpha = i, i+1, \ldots, j-1$ are optimal.

(b) If $t_n > T^*$ go to STEP 3.

(c) If $t_n < T^*$ and there exists an $s_{ij} \neq 0$, go to STEP 4.

STEP 3: Solve the linear programming problem (ELP2). Go to STEP 2.

STEP 4: Find $s_{ij} > 0$ such that d_{ij} is a maximum. Then put

$$\hat{s}_{ij} = \text{Max}\{(s_{ij} - R_{\hat{a}}(T^* - t_n)) ; 0\}$$

$$\hat{x}_{ij(\hat{a})} = M_{ij} - \hat{s}_{ij}.$$

If $t_n = T^*$ terminate the algorithm. The optimum has been reached. Otherwise $t_n < T^*$ and there exists an $s_{ij} > 0$, repeat STEP 4.

The workings of the procedure are now illustrated by an example. Consider a project with parameters as specified in Table III. In addition, a target date has been set at 38 days from the time of commencement of the project ($T^* = 38$).

A penalty cost of $C^* = \$100$ is assessed for each day beyond this target.

Table III

Activity (i,j)	Normal Cost/ Manday ($) (c_{ij})	Overtime Cost/ Manday ($) (d_{ij})	Required Mandays (M_{ij})
(1,2)	6	9	45
(2,3)	8	12	24
(2,4)	8	12	50
(2,5)	6	9	90
(4,6)	4	6	36
(5,6)	10	15	12
(5,7)	6	9	56
(6,8)	6	9	24
(7,8)	8	12	40
(8,10)	10	15	20
(9,10)	6	9	21

It is also specified that only 21 mandays of overtime may be employed to attempt to meet the due date. The problem is to schedule normal and overtime personnel to minimize the total cost of completing the project. The resource restrictions R_i given by Table IV must also be met. The project can be represented by the network of Figure 12.

Table IV

Event (i)	1	2	3	4	5	6	7	8	9
Resources (R_i) Available in $[t_i, t_{i+1})$	15	6	10	10	6	8	8	10	7
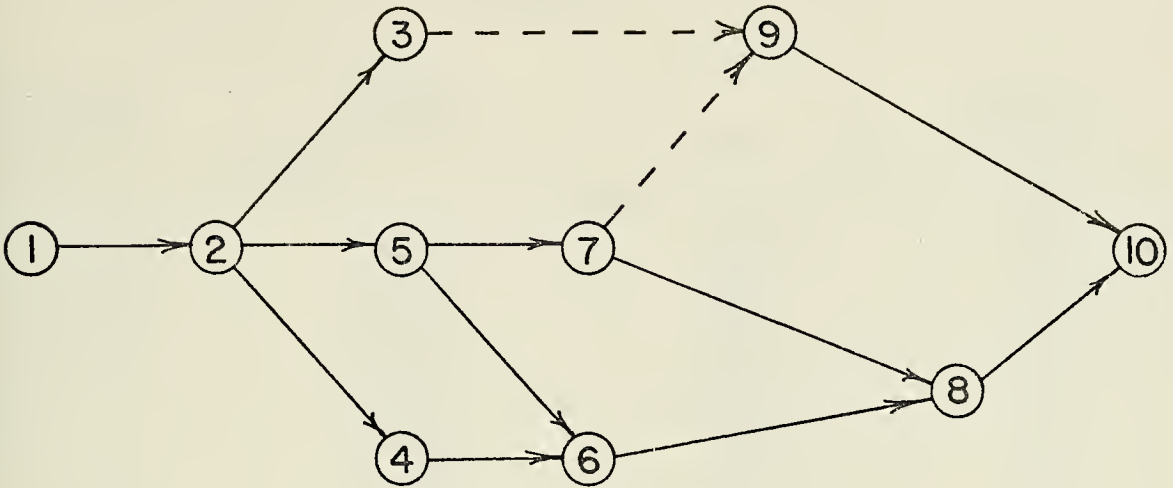
Figure 12

Following the algorithm, all overtime is set equal
to zero and the minimum project duration is found using
STEP 2. A summary of this procedure appears in Table V.
The minimum value of $1/R_\alpha$ for each activity is marked with
an asterisk.

The event times are then found using (IV-D-13).
They are: $t_1 = 0$, $t_2 = 3$, $t_3 = 7$, $t_4 = 12$, $t_5 = 24$, $t_6 = 26$,
$t_7 = 33$, $t_8 = 41$, $t_9 = 43$, and the project duration $t_{10} = 46$.
The project duration of 46 days exceeds the due date by 8
days. Then, as specified by the algorithm, the linear pro-
gramming problem (ELP2) is solved. The solution to this
problem as shown in Table VI, consists of putting $s_{23} = 21$
mandays (the maximum permissible overtime). Since $M_{23} = 24$
there are 3 mandays left to be allocated during the normal
working day. All other resource allocations remain the same.

100

Table V

Activity (i,j)	State Variable $x_{ij(\alpha)}, \alpha=i,\ldots,j-1$	$1/R_\alpha$	Value of $x_{ij(\alpha)}$
(1,2)	$x_{12}(1)$	1/15*	45
(2,3)	$x_{23}(2)$	1/6*	24
(2,4)	$x_{24}(2)$	1/6	0
	$x_{24}(3)$	1/10*	50
(2,5)	$x_{25}(2)$	1/6	0
	$x_{25}(3)$	1/10	0
	$x_{25}(4)$	1/10*	90
(4,6)	$x_{46}(4)$	1/10*	36
	$x_{46}(5)$	1/6	0
(5,6)	$x_{56}(5)$	1/6*	12
(5,7)	$x_{57}(5)$	1/6	0
	$x_{57}(6)$	1/8*	56
(6,8)	$x_{68}(6)$	1/8	0
	$x_{68}(7)$	1/8*	24
(7,8)	$x_{78}(7)$	1/8*	40
(8,10)	$x_{8,10}(8)$	1/10*	20
	$x_{8,10}(9)$	1/7	0
(9,10)	$x_{9,10}(9)$	1/7*	21

Table VI

(i,j)	c_{ij}	d_{ij}	C^*/R_α	h_{ij}
(1,2)	6	9	6.67	- 3.67
* (2,3)	8	12	16.67	-12.67
(2,4)	8	12	10.00	- 6.00
(2,5)	6	9	10.00	- 7.00
(4,6)	4	6	8.33	- 6.33
(5,6)	10	15	16.67	-11.67
(5,7)	6	9	12.50	- 9.50
(6,8)	6	9	12.50	- 9.50
(7,8)	8	12	12.50	- 8.50
(8,10)	10	15	10.00	- 5.00
(9,10)	6	9	14.29	-11.29

The minimum total cost is $3386 consisting of $2624 in normal
labor costs, $252 for overtime, and a penalty of $510 for
exceeding the target date by 5.1 days. The total cost of
minimizing duration only is $2792 + $800 = $3592. The
optimal time-oriented network and the final resource profile
are illustrated in Figure 13. Note that all of the resource
available is applied to one activity at a time. This was
done to ensure integral numbers of men.

F. APPLICATION OF THE PROCEDURE

The solution procedure developed in this chapter depends
heavily on the four assumptions stated in Section B. Assump-
tion 1, the event ordering requirement and assumption 2, the
resource availability assumption cause the greatest restric-
tions on the application of the procedure to a realistic
problem. It is worthwhile to examine these assumptions more
closely and to suggest methods for reducing the limitations
caused by these assumptions.

1. Ordering the Events

a. Role of the Event Ordering Assumption

Assumption 1 requires that each of the events in
the project be numbered 1,2,...,n so that their attainment
times are nondecreasing, $t_1 \leq t_2 \leq \ldots \leq t_n$. This requirement
can be very restrictive on the application of the procedure
to a real problem.

In order to illustrate a case in which this
assumption is severe, consider a project represented by a

102
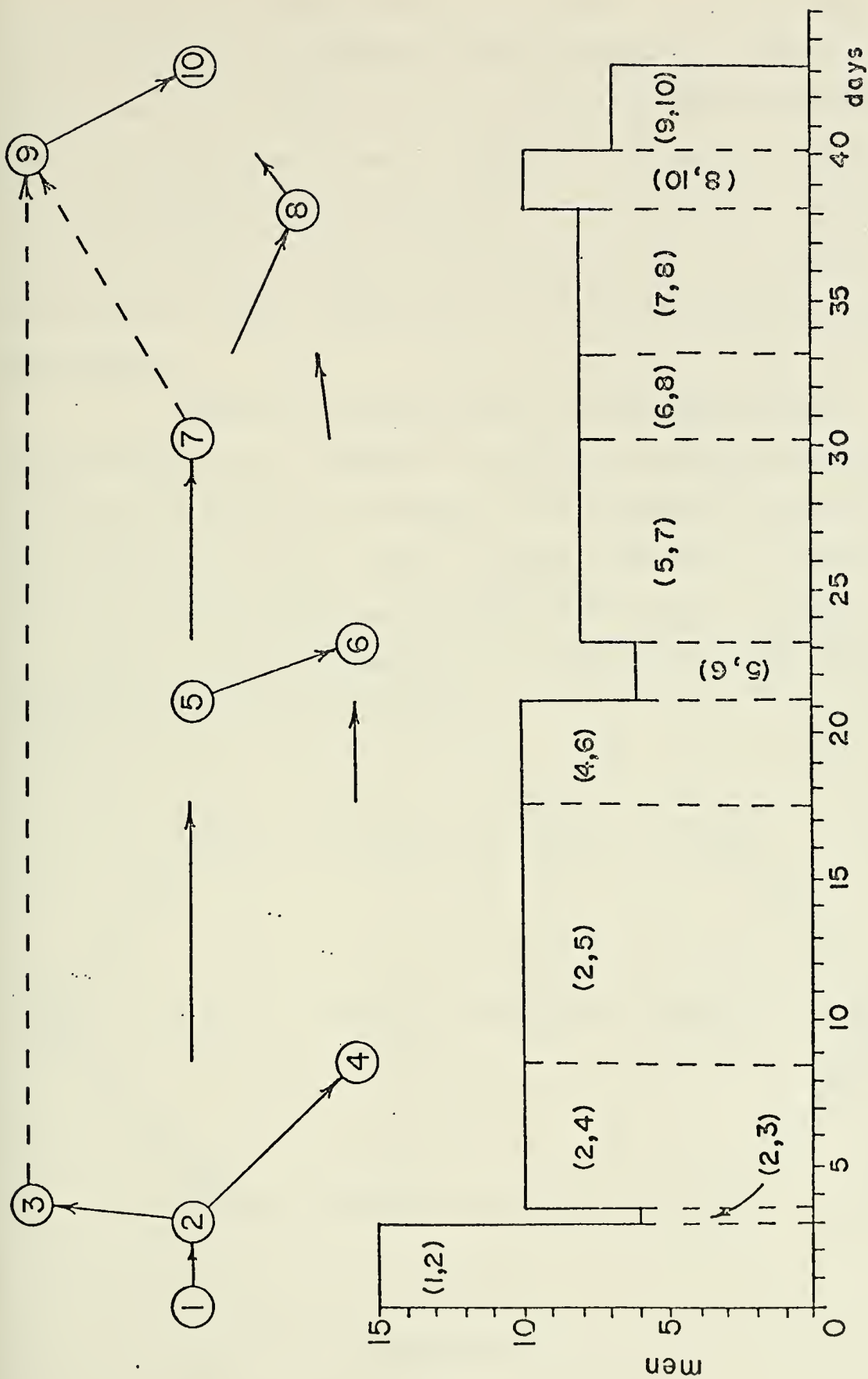
figure 13

103

series-parallel network such as in Figure 14. The network
form of the n job m machine job shop scheduling problem has
this structure. Suppose each event must be numbered before
the solution procedure can be applied. The optimal solution
so obtained might be meaningless in this situation. The
objective of the job shop scheduling problem is, in fact, to
find the best event numberings and their associated attain-
ment times.

There are some projects, on the other hand,
whose events can be numbered naturally so that the assumption
would be automatically satisfied. The network of Figure 15
illustrates this case. Most projects, however, fall some-
where in between these two extremes. For those projects,
the extent to which assumption 1 is restrictive very much
depends on the network structure.

Figure 14

104

Figure 15

b.   Enumeration of Event Orderings

In order to state that a given solution to a given network cost minimization problem is optimal, it should remain optimal for any numbering of the events. To find this optimal solution, then it is necessary to obtain every possible event numbering scheme either implicitly or explicitly. This is no simple task for most real project networks.

A method is now presented which yields every feasible event ordering. A feasible event ordering is a numbering of the events so that if an activity (i,j) exists then $i < j$. Fulkerson [31] provided an algorithm for finding some feasible ordering for any acyclic network. There may be many feasible orderings for a given network and the present task is to discover all of them.

The procedure for generating all the feasible event orderings uses the notion of disjunctive activities.

·Several authors [2], [3], [4], [15], [16], [34], [70], [71]
have used disjunctive graphs (networks composed of both
normal and disjunctive activities) to provide solution pro-
cedures to various scheduling problems.

A disjunctive activity (arc), (i,j) is an activity
which can either be directed from event i to event j or from
event j to event i but not both. Figure 16 illustrates the
graphical representation of the disjunctive activity (i,j).
The event ordering generation procedure consists of first
connecting all nodes either by existing activities, chains
of existing activities, or disjunctive activities. All
orientations of the disjunctive activities which do not yield
cycles are then generated. For any fixed acyclic orientation
of the disjunctive activities, Fulkerson's event numbering
procedure [31] can be utilized to obtain the associated
unique event numbers

Figure 16

The first step is to label (number) all events
that can be numbered uniquely. There are at least two label-
led events. The source node is labelled 1, the sink node n.
If all events have been labelled, the procedure can be ter-
minated and only one feasible event ordering exists. For
most realistic cases, there will be a set of nodes that are

not labelled. Call this set U. In this set connect each
pair of events i and j by an arrow leading from i to j if in
the project network activity (i,j) exists or if there is a
chain of activities from i to j. All of the remaining events
in U are then connected pairwise by disjunctive activities.

From the last labelled node scan all arrows (i,j)
and make every possible orientation of the disjunctive
activities connecting those nodes j. Call the set of these
scanned nodes S. If fixing a disjunctive activity in a
particular direction generates a cycle, the reverse orienta-
tion must be established. Each fixed orientation may or may
not generate a new numbered node. If a new node is labelled,
the procedure is repeated scanning arrows emanating from the
labelled node. If a new node is not labelled all arrows
emanating from the noeds in S are scanned and the procedure
is repeated. Each fixed orientation of a disjunctive activ-
ity describes the unique ordering of a pair of events. If
the number of disjunctive activities in the network is p
then the number of event orderings is at most 2^p. Many of
these generated orderings may be infeasible.

The procedure is now illustrated by a simple
example. Consider the project network of Figure 17(a).
Following the procedure described above, label all nodes
that can be numbered uniquely. Only nodes 1 and 6 can be
labelled uniquely. U is the set of unlabelled nodes and is
represented by the dotted line in Figure 17(b). The unlabel-
led nodes are temporarily marked a-d for identification
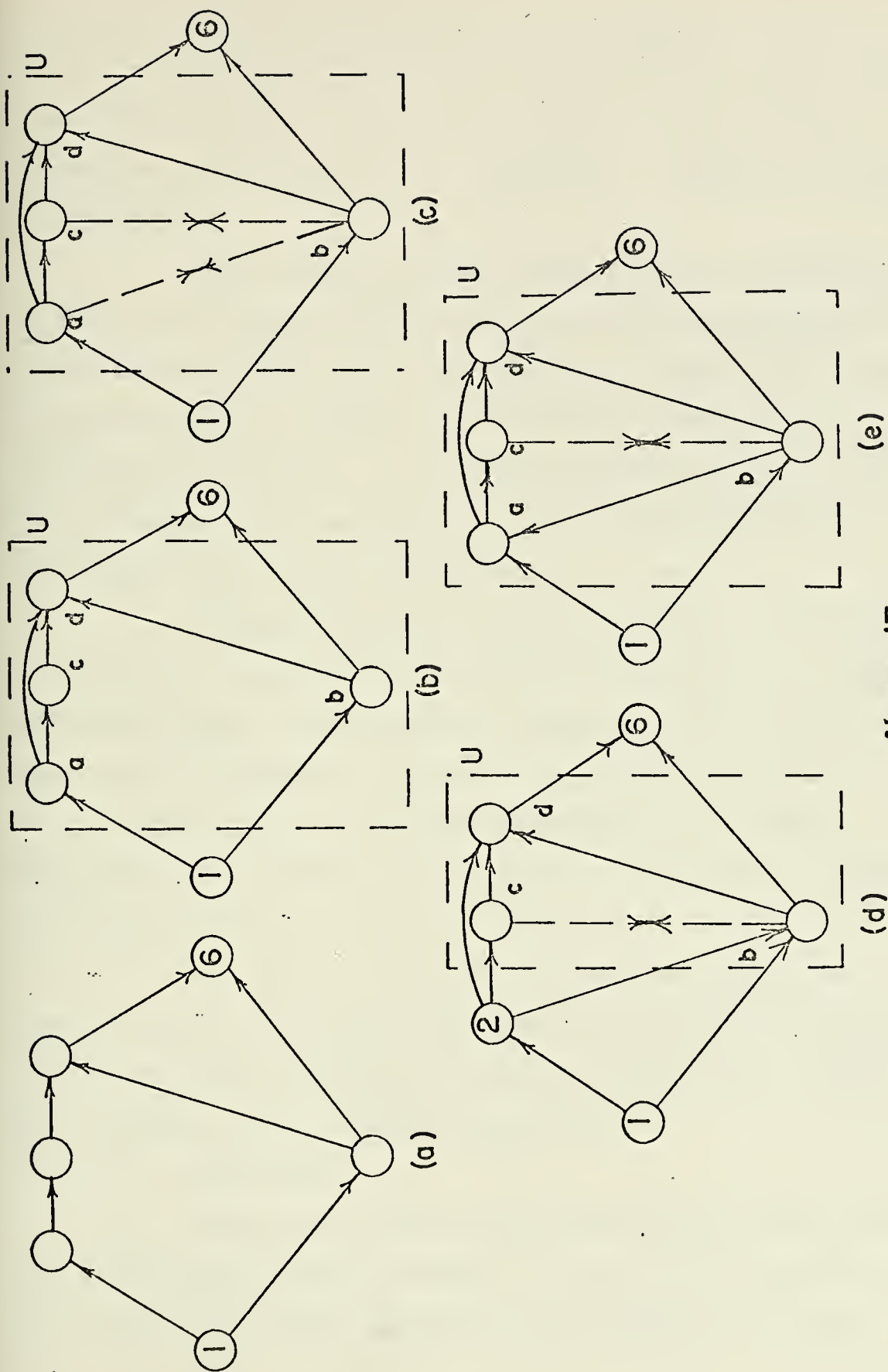
figure 17

purposes. Then the heavy arrow (a,d) represents the only
chain in U. Then in Figure 17(c) the remaining node pairs
(a,b) and (c,b) are connected by disjunctive activities.
There are then, at most, $2^2 = 4$ different orderings of the
nodes in U.

Node 1 was the last labelled node so the nodes
a and b are placed in S and the two possible orientations of
disjunctive activity (a,b) are examined. These two orienta-
tions are shown in Figure 17(d) and (e). Note that the
orientation in Figure 17(d) permits labelling of node 2 and
hence changes the composition of U while the orientation
of Figure 17(e) leaves U unchanged.

Branching from the network of Figure 17(d) we
further obtain the networks of Figure 18(a) and (b). Like-
wise from Figure 17(e) we obtain Figures 18(c) and (d).
Note that the network of Figure 18(d) yields the cycle
b-a-c-b hence yields an infeasible ordering. The three
possible event ordering combinations of this project network
are shown in Figure 18(a)-(c).

In order to find the minimum total cost for a
project with this network configuration it would be necessary
to use the cost minimization algorithm three times, once for
each feasible event ordering combination.

c. Heuristics

The size of the project network may prevent the
use of the event ordering method. If so, it is necessary to
pick one or more event ordering combinations and use them

figure 18

110
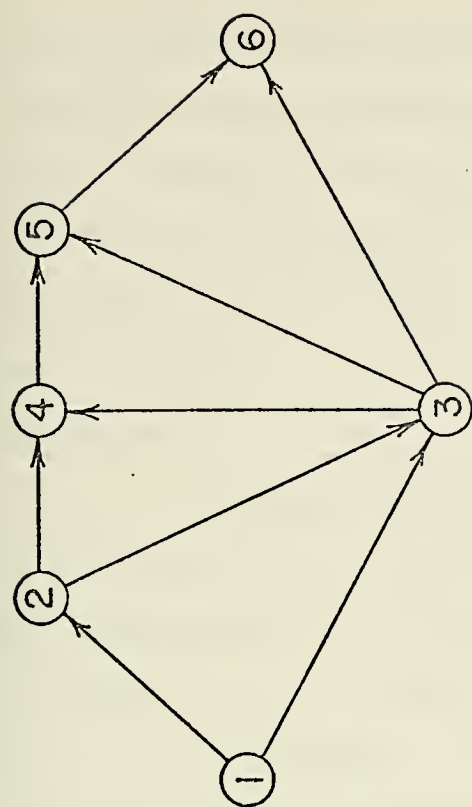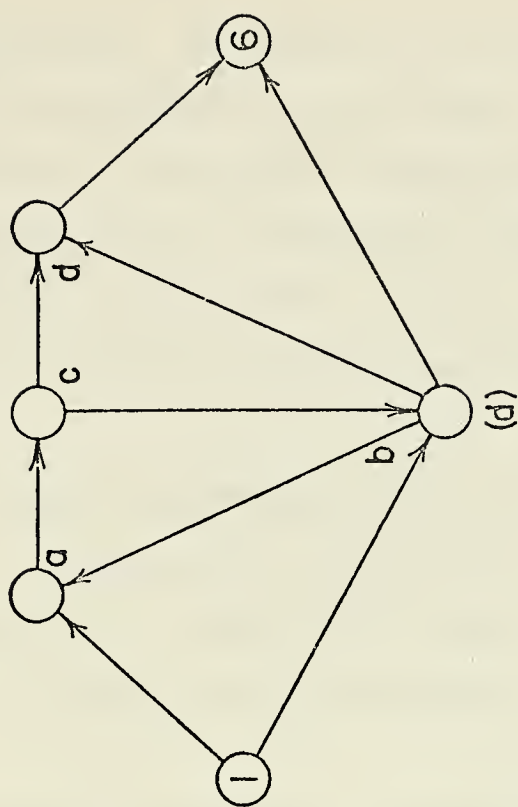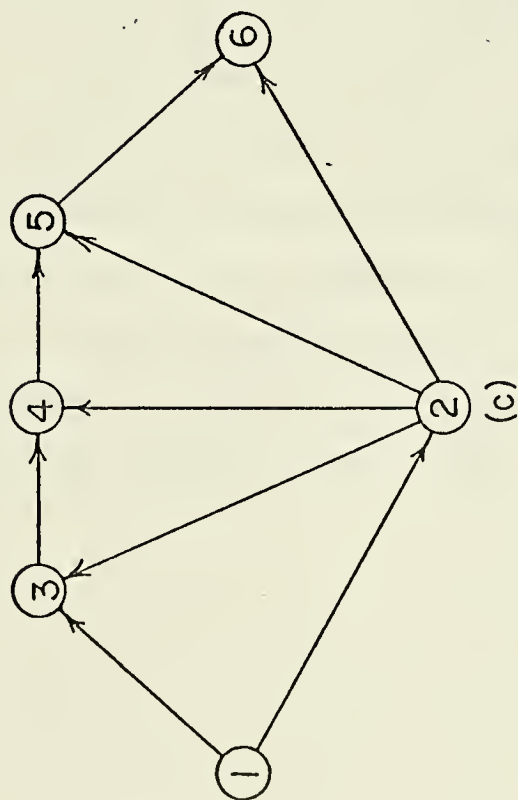
with the cost minimization algorithm. The selection of an ordering combination might be accomplished by using some rule of thumb to order the events. One possible heuristic rule might be to order the events based on increasing total M_{ij} values of activities incident to the events. In any case, a heuristic scheme cannot guarantee the discovery of the optimal solution.

An alternative to using a heuristic might be to randomly generate the event orderings while ensuring that the destination of each activity has a higher event than its origin. This procedure also cannot ensure optimality.

d. An Example

The cost minimization algorithm was programmed in FORTRAN IV for the Naval Postgraduate School's computer, an IBM 360/67. The 70 feasible orderings for the example of Section IV-E-2 were determined by hand using the event ordering procedure suggested in Section b above. The minimum total cost for each of these possibilities was then determined using the algorithm. The minimum total cost computed in this manner was $2378. The feasible ordering combination that yields this cost is shown in Figure 19. This is also the ordering generated by the suggested heuristic rule. The maximum total cost was $3825.
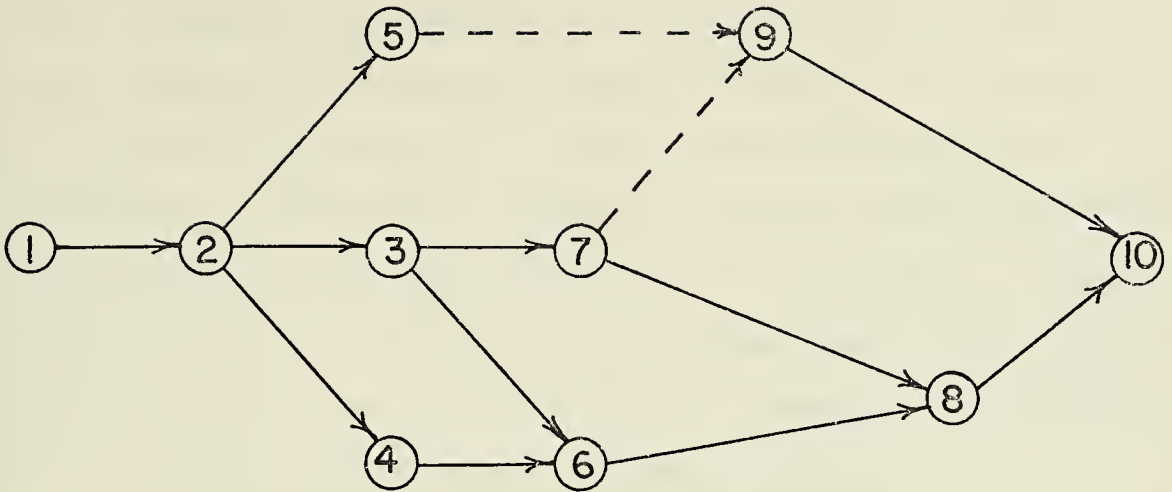
Figure 19

The cost minimization routine was performed for groups of 10 or more possible orderings within a DO loop allowing efficient use of compilation time. The average time for computing the minimum total cost for each event ordering combination was 0.53 seconds. The program required 54×10^3 bytes of storage for this 10 event 13 activity problem. A more detailed description of computation appears in Appendix B.

2. Resource Availability

    a. Role of the Resource Availability Assumption

        The planning of real shipyard projects must normally be done subject to scarce resources. The resource constraints must usually be represented in a form such as in Chapter III of this thesis. Assumption 2 of Section B of this chapter requires that the resources availability break-points be at the event times rather than at fixed points in

time. Although this assumption made the solution of the single resource problem very simple, it might not be valid for a realistic project. If, after performing the cost minimization, the resource availability break points coincide with the appropriate event times then the procedure is valid for that particular project. Most of the time, however, this will not be the case. Very often some resource constraints will be violated while other resources will be idle.

The effects of this assumption might be reduced by dividing some activities into two or more sub-activities in series. The total manday requirements for the sub-activities are required to equal the total activity manday requirements. This assumption might also be more acceptable if shipyard policy allows resources to be transferred among various projects in the shipyard for short periods of time. Idle resources from some projects could be shifted to resource constraint violations in other projects.

b. Approximate and Exact Solutions — An Example

Consider a project represented by the network of Figure 20(a). The numbers on the activities are manday requirements. Suppose the problem is to minimize the project duration subject to resource restrictions over time illustrated by the project resource profile of Figure 20(b). The cost of minimization algorithm with overtime and penalty omitted from the model yields the minimum duration. Suppose now that the resource profile used in the algorithm is given by Table VII. The algorithm yields a minimum duration of

113

(a)

(b)

figure 20

114
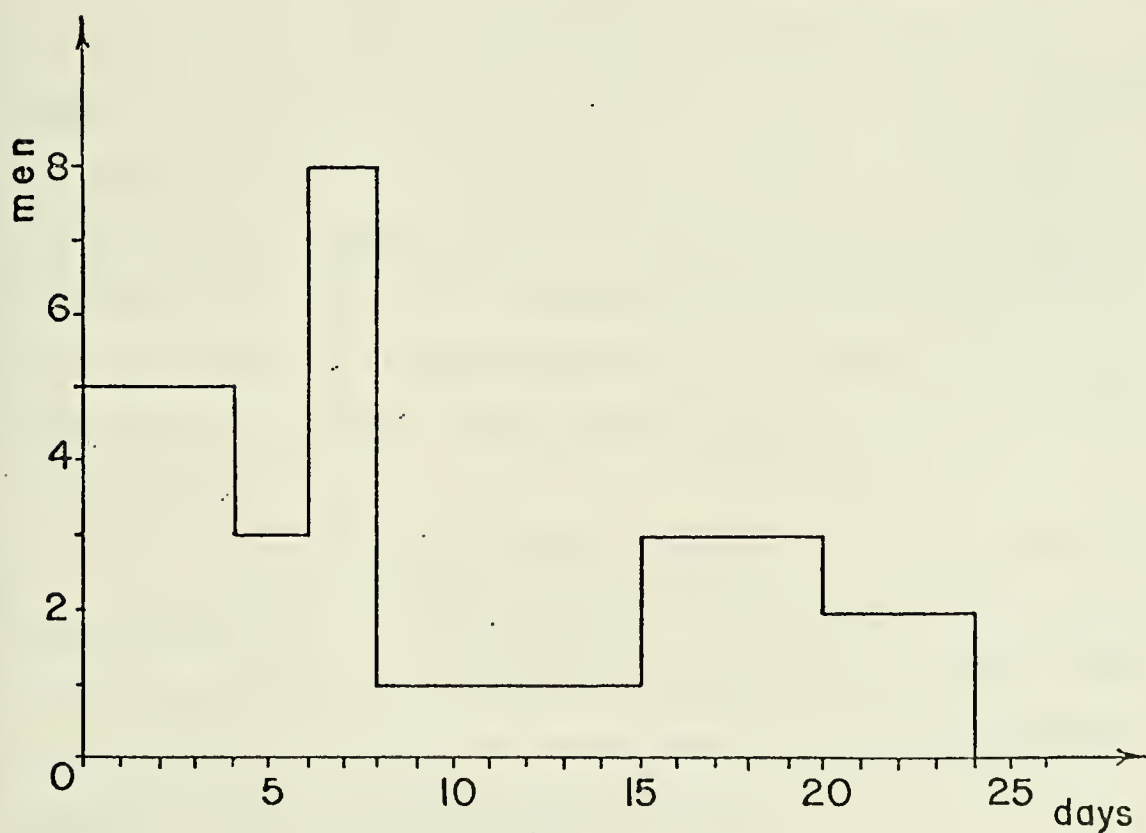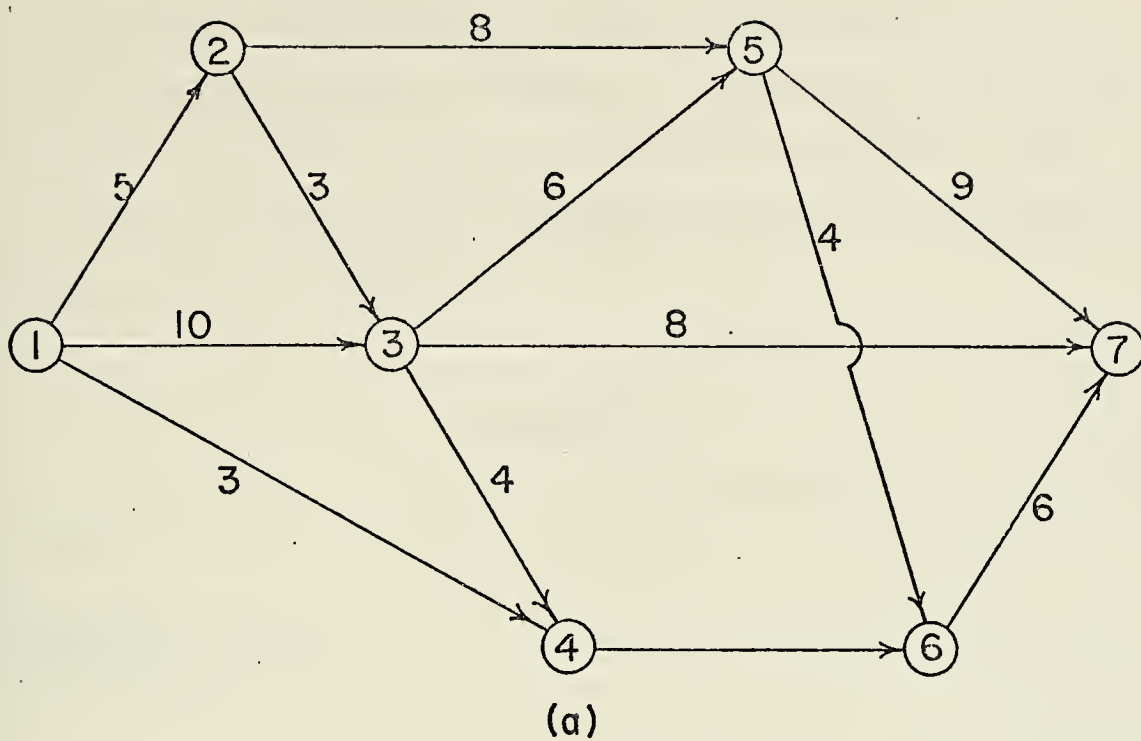
15 5/8 days. The associated project resource profile is
shown as a dashed line on Figure 21. The solid line in this
figure is the actual resource availability profile. Note
that the cost minimization algorithm results in an approxima-
tion to actual conditions.

Table VII

Event (i)	1	2	3	4	5	6
Resource Availability (R_i)	5	3	8	1	3	2

Now suppose some activities are broken into sub-
activities with manday requirements divided among them.
Suppose the activities (2,3), (2,5), (3,7), (4,6), and (4,7)
are each divided into two serial activities. The revised
network and appropriate manday requirements are illustrated
in Figure 22. Now if the resource availabilities of Table
VIII are used, the minimum duration is 22 days and the assoc-
iated resource usage profile coincides with the actual resource
availability profile.

The purpose of this example is to show that for
some networks there is a division of activities that can be
made to more closely approximate reality. Judicious separa-
tion of the activities can often result in a closer approx-
imation to an existing resource profile.
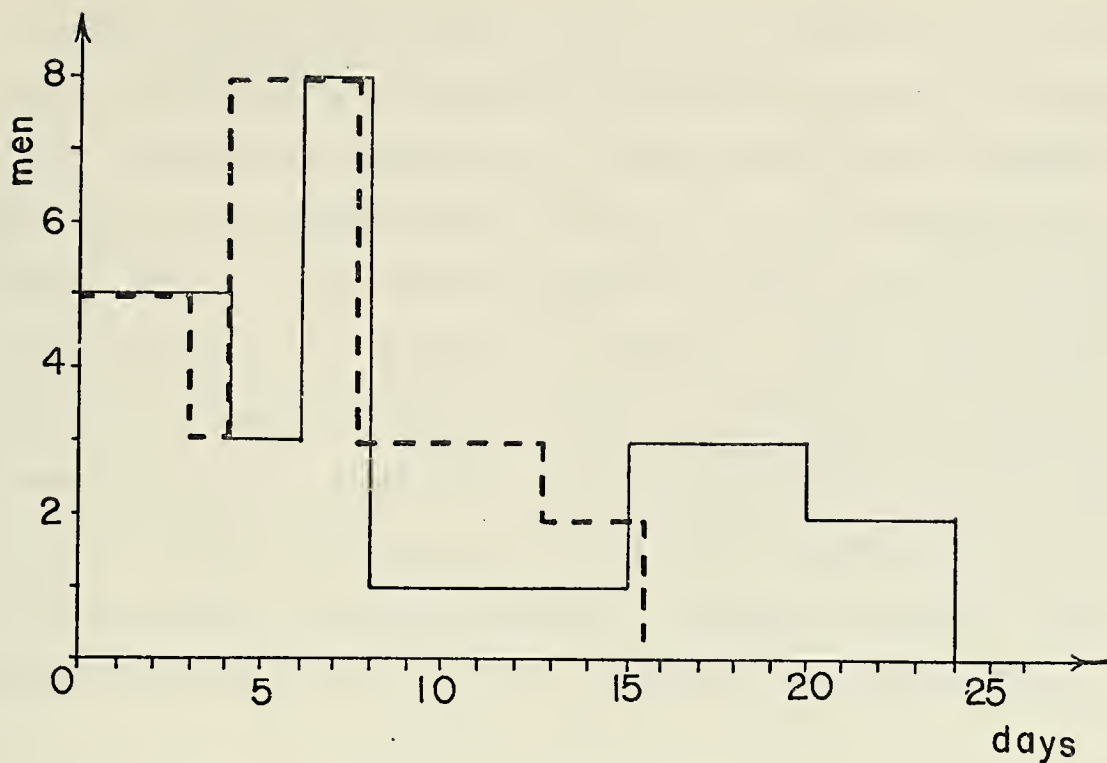
Figure 21

Table VIII

i		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
R_i		5	5	5	5	3	3	3	8	8	1	1	1	3	3	3	3	2

G. MULTIPLE RESOURCES

The development in this chapter has applied only to the case where a single resource is constrained over time. If there are several resource types whose availabilities are limited, other approaches must be taken.

116

Some authors [68], [72], [81] have succeeded in developing
solution procedures to similar scheduling problems by stating
a very restrictive assumption. These formulations require
that each of the event times t_1, t_2, ..., t_n be fixed and
known. That is, the starting times of each of the activities
in the project are specified in advance. Each of the papers
mentioned attacked a resource leveling problem where several
resource types were involved. These approaches are interesting
and are briefly described in this section. The total cost
model developed in this chapter is then examined after the
addition of the fixed activity starting time assumption.
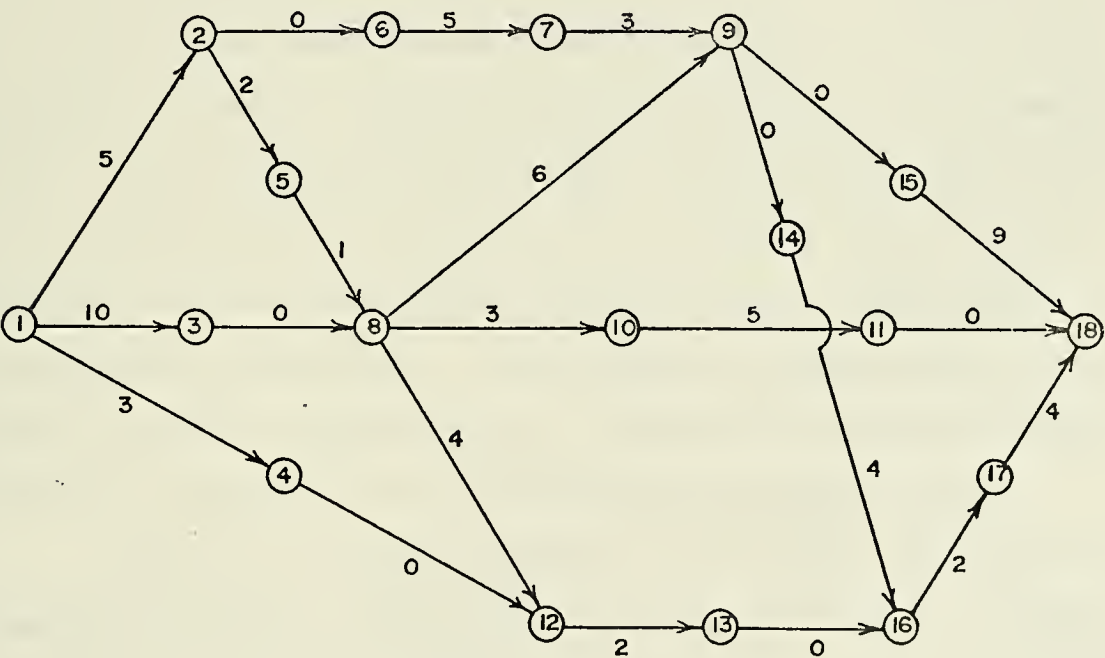
Figure 22

1. A Hydrostatic Model

A Soviet operations research analyst, B. S. Razumikhin, developed a hydrostatic model to describe a resource leveling problem [72]. For this problem a project is made up of activities which require a fixed number of resource-time units for completion. This condition, just as in the total cost model, can be represented by

$$(IV-G-1) \qquad \sum_{\alpha=i}^{j-1} r_{ij}^{k}(t_\alpha)(t_{\alpha+1} - t_\alpha) = M_{ij} \; ; \quad \forall (i,j) \epsilon N.$$

This system of equations is represented by Razumikhin with a fluid mechanic model. Each resource allocation $r_{ij}^{k}(t_\alpha)$ is the height of a cylinder whose width is $t_{\alpha+1} - t_\alpha$ and whose thickness is unity. The cylinder contains an incompressible fluid. For a single resource type k and a set of activities (i,j) which require the use of this resource there are n-1 cylinders of this type placed side by side. This can be illustrated by Figure 23. The network of Figure 23(a) requires a single resource type. Some set of resource allocations are represented for this network in Figure 23(b).

The fluid in the cylinders for a single activity can communicate freely. The total volume of fluid for cylinders of activity (i,j) must remain constant at M_{ij} so if the height of one cylinder say $r_{13}(t_1)$ decreases then the cylinder height $r_{13}(t_2)$ must increase. The cylinder sides (event times) t_1, t_2, ..., t_4 are fixed. Razumikhin showed that if the cylinder heights (resource allocations) are varied until
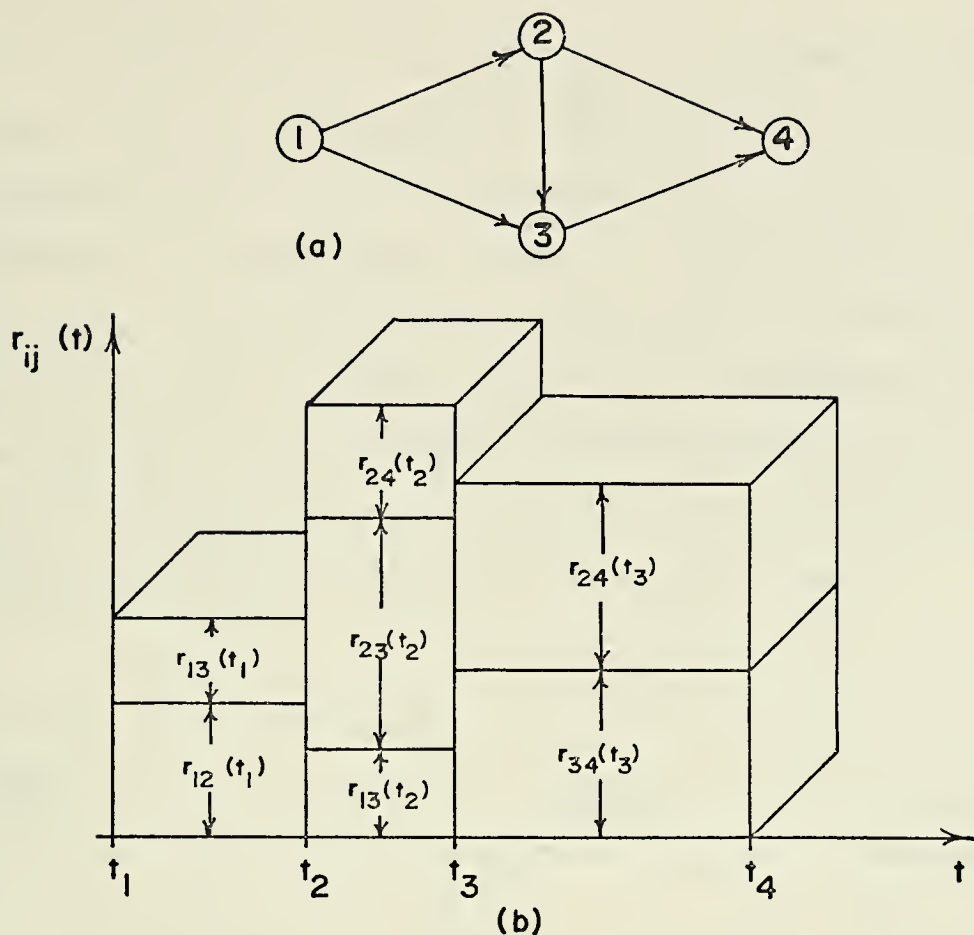
Figure 23

the minimum potential energy is attained, the result is the
same as minimizing the mean square deviation of resources
from a constant. Or, this results in a leveling of the
resources.

If K different resources are required for the project
then there are K independent cylinder systems such as in
Figure 24. Razumikhin provided a method of successive

approximations for finding the minimum potential energy for
this system.

It was then stated that if the intermediate event
times t_1, t_2, ..., t_{n-1} were allowed to vary, then the optimal
resource allocation in the project would correspond to equi-
librium in the hydrostatic model. If the cylinder walls
t_1, t_2, ..., t_n are movable, equilibrium corresponds to the
situation where the system has minimum potential energy and
the pressure on each cylinder side wall is the same. Noticing
this, Razumikhin presented another successive approximation
technique to allow variation of the event times.
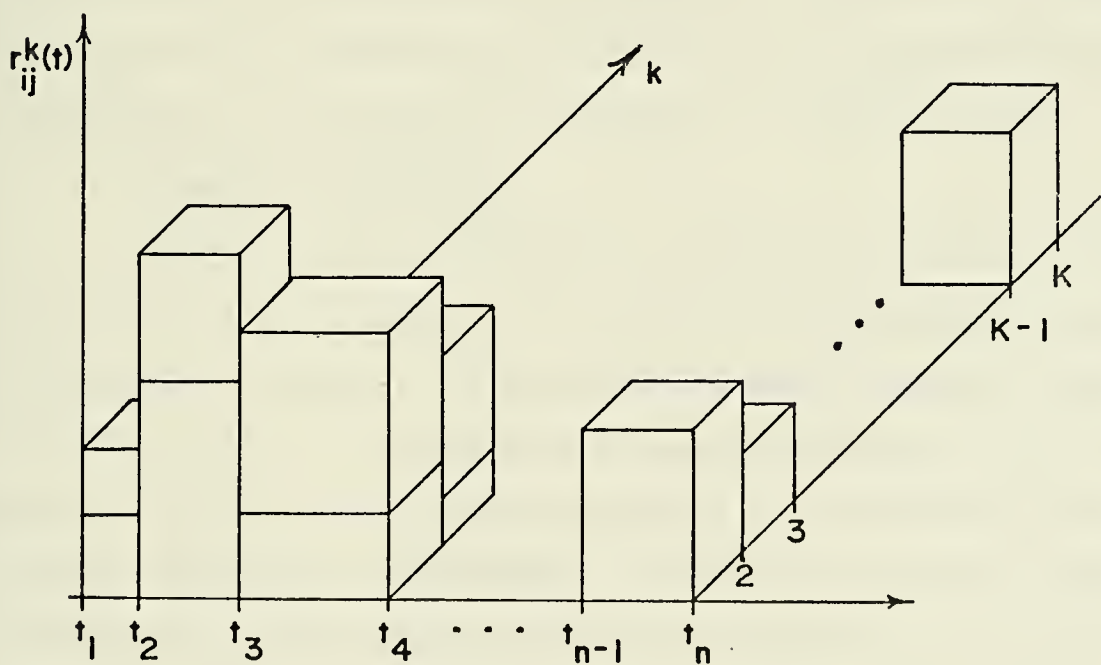
Figure 24

The procedure is summarized briefly below.

STEP 1: Fix t_1, t_2, ..., t_n and find the resource allocations $r_{ij}^k(t)$ which yield minimum potential energy of the cylinder system. Go to STEP 2.

STEP 2: Vary t_1, t_2, ..., t_{n-1} (and consequently $r_{ij}^k(t)$) so that the force exerted by the fluid on each cylinder wall is constant. Then return to STEP 1 with these values of t_i. Terminate the procedure when there is little change in potential energy and force at succeeding iterations.

In a later paper [74], Razumikhin extended this fluid mechanic model to represent the problem of minimizing project duration with a bounded total expenditure of resources. No computational considerations are given in either paper.

2. Dynamic Programming

Petrovic [68] approached a resource leveling problem using dynamic programming. The problem attacked had constraints on resources similar to those of the minimum duration model of Chapter III. The event times for each activity were fixed and this yielded a discrete multistage system suitable for solution by dynamic programming. Finding the optimal resource allocations at each stage with multiple resources is an extremely difficult task in itself. Petrovic outlined a few techniques for reducing the amount of computation needed to perform this.

3. Quadratic Programming

Beale's quadratic programming method [6] was applied by Voronov and Petrushinin [81] to the same resource leveling problem that Razumikhin approached. The problem with the objective the minimization of the mean square deviation of resources from a constant can be written as

(IV-G-2)   Minimize $\displaystyle\sum_{\alpha=1}^{n-1} \left[\sum_{(i,j)\epsilon S_k} r_{ij}^k(t_\alpha) \right]^2 (t_{\alpha+1} - t_\alpha)$

Subject to

(IV-G-3) $\displaystyle\sum_{\alpha=1}^{n-1} (t_{\alpha+1} - t_\alpha) r_{ij}^k(t_\alpha) = M_{ij} \; ; \; \forall (i,j)\epsilon N$

with

(IV-G-4) $r_{ij}^k(t_\alpha) \begin{cases} \equiv 0 \text{ for } \alpha < i \text{ or } \alpha \geq j \\ \geq 0 \text{ for } i \leq \alpha < j. \end{cases}$

When the event times t_1, t_2, ..., t_n are fixed the above problem becomes a quadratic programming problem with the additional constraints (IV-G-4). The authors applied the Kuhn-Tucker conditions and extended the Beale quadratic programming algorithm to solve the problem. This is an interesting method for solving the resource leveling problem because it is very simple. The assumption that event times are known in advance, however, is very restrictive and limits the acceptability of this procedure.

122

4. A Generalized Transportation Problem

The total cost problem (TCP2) of Section E of this chapter can be greatly simplified if the fixed event time assumption mentioned above is made. The total cost problem is stated again for convenience. The problem is

(IV-G-5) Minimize $\left[\sum_{(i,j)\epsilon N} (d_{ij}^k - c_{ij}^k) s_{ij}^k + C^*(t_n - T^*) \right]$

Subject to

(IV-G-6) $\sum_{\alpha=i}^{j-1} r_{ij}^k(t_\alpha)[t_{\alpha+1} - t_\alpha] + s_{ij}^k = M_{ij}$; $\forall (i,j)\epsilon N$

(IV-G-7) $\sum_{(i,j)\epsilon P_q} r_{ij}^k(t_q) \leq R_q^k$; $\begin{array}{l} k = 1, 2, \ldots, K \\ q = 1, 2, \ldots, n-1 \end{array}$

(IV-G-8) $\sum_{(i,j)\epsilon N} s_{ij}^k \leq B$

(IV-G-9) $s_{ij}^k \geq 0, \ t_i \geq 0, \ t_1 = 0, \ r_{ij}^k(t) \begin{cases} \geq 0 \text{ for } t_i \leq t < t_j \\ \equiv 0 \text{ for } t < t_i \\ \quad \text{or } t \geq t_j \end{cases}$

(IV-G-10) $C^*(x) = \begin{cases} C^*x \text{ for } x \geq 0 \\ 0 \text{ for } x < 0. \end{cases}$

Suppose now that $t_1, t_2, \ldots, t_n$ are fixed and the values are known. The penalty cost term then drops out of the objective function and each of the time intervals $t_{\alpha+1} - t_\alpha$; $\alpha = 1, \ldots, n-1$ are constants a_α. The problem can then be written as

123

(IV-G-11)   Minimize $\displaystyle\sum_{(i,j)\epsilon N} (d_{ij}^k - c_{ij}^k)s_{ij}^k$

Subject to

(IV-G-12) $\displaystyle\sum_{\alpha=i}^{j-1} a_\alpha r_{ij\alpha}^k + s_{ij}^k = M_{ij} \; ; \quad \forall (i,j)\epsilon N$

(IV-G-13) $\displaystyle\sum_{(i,j)\epsilon P_q} r_{ijq}^k \leq R_q^k \; ; \quad \begin{array}{l} k = 1, 2, \ldots, K \\ q = 1, 2, \ldots, n-1 \end{array}$

(IV-G-14) $\displaystyle\sum_{(i,j)\epsilon N} s_{ij}^k \leq B$

(IV-G-15) $s_{ij}^k \geq 0, \; r_{ij}^k \begin{cases} \geq 0 \text{ for } i \leq \alpha < j \\ \equiv 0 \text{ for } \alpha < i \text{ or } \alpha \geq j. \end{cases}$

The problem (IV-G-11)-(IV-G-15) is a generalized transportation problem [37, p. 314]. The constraints (IV-G-15) on $r_{ij\alpha}^k$ can be taken care of by assigning an arbitrarily large cost to the variables $r_{ij\alpha}^k$ when $\alpha < i$ or $\alpha \geq j$.

If the time differences a_α are of unit length the problem is placed in a form similar to that given in Chapter III. Since each of the event times are known, however, there is no need to adjoin constraints involving zero-one variables.

H. SUMMARY

A nonlinear programming formulation for the total cost problem was given in this chapter. In its most general form,

124

the model is extremely complex. When additional assumptions
are made, however, simple solution procedures can be applied.

The cost minimization algorithm for a single constrained
resource is very simple and can compute the minimum cost of
a project very quickly (See Appendix B). The limitation here
is that the events must be ordered in a particular way. This
may or may not impose a severe restriction on the scheduler,
depending on the project in question. Finally, if the
manager is willing to specify activity starting times in
advance, the problem is again very simple.

V. FIXED RESOURCE PROFILES

A. INTRODUCTION

The scheduling of resource constrained projects to satisfy any objective function is an extremely difficult task. The combinatorial nature of these scheduling problems usually requires the statement of simplifying assumptions permitting the use of known solution methods. The nature of the assumptions and the restrictions caused by them determine whether the solution obtained compares favorably with reality or not.

A class of resource constrained project scheduling problems that for many problems is quite realistic is that of projects composed of activities with fixed resource profiles. An activity resource profile is a graph of the resources required to perform the activity over time. An example of an activity resource profile is shown in Figure 25. In the example, r_1 units of resource must be used on activity (i,j) at time t_i and must be employed until time $t_i + d_1$. Time interval d_1 is the duration over which r_1 units must be employed on activity (i,j). The activity is completed when each of the required resource units has been used for the appropriate lengths of time.

A special case of the fixed resource profile which also can represent a realistic use of manpower is the constant resource profile. This type of activity resource profile is represented in Figure 26. To complete an activity it is necessary to employ r_{ij} units of the required resource for

a length of time equal to T_{ij}, the activity duration. If
the manager is willing to specify how many men will work on
an activity and for how long then some existing scheduling
procedures can be used. This specification amounts to
constructing a constant resource profile for each activity.
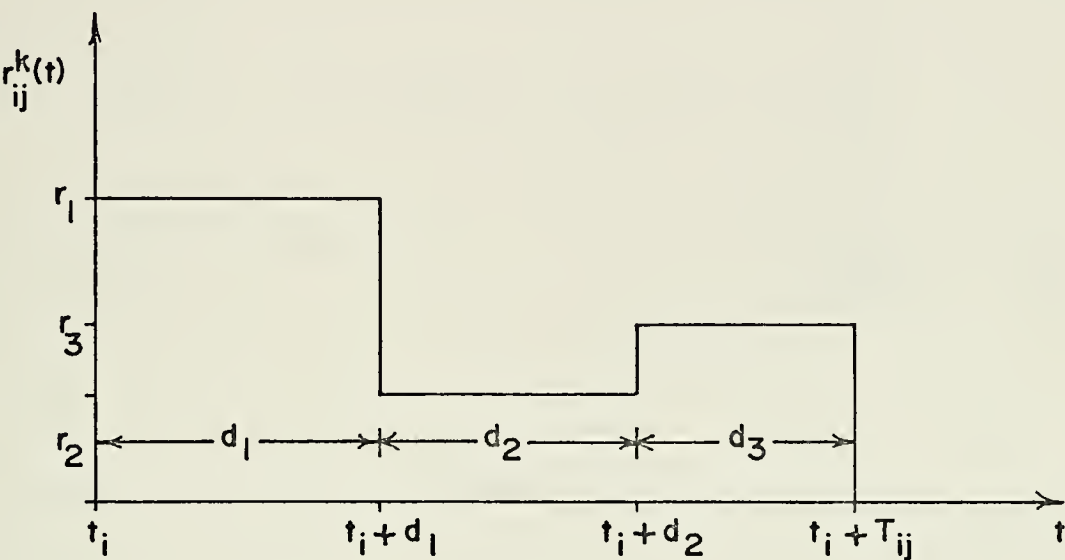
Figure 25

In this chapter, the shipyard scheduling problem of
minimizing project duration with fixed activity manday
requirements and restrictions on available resources is
examined. When certain conditions exist and when fixed
activity profiles are adopted, some existing scheduling
methods can be employed.
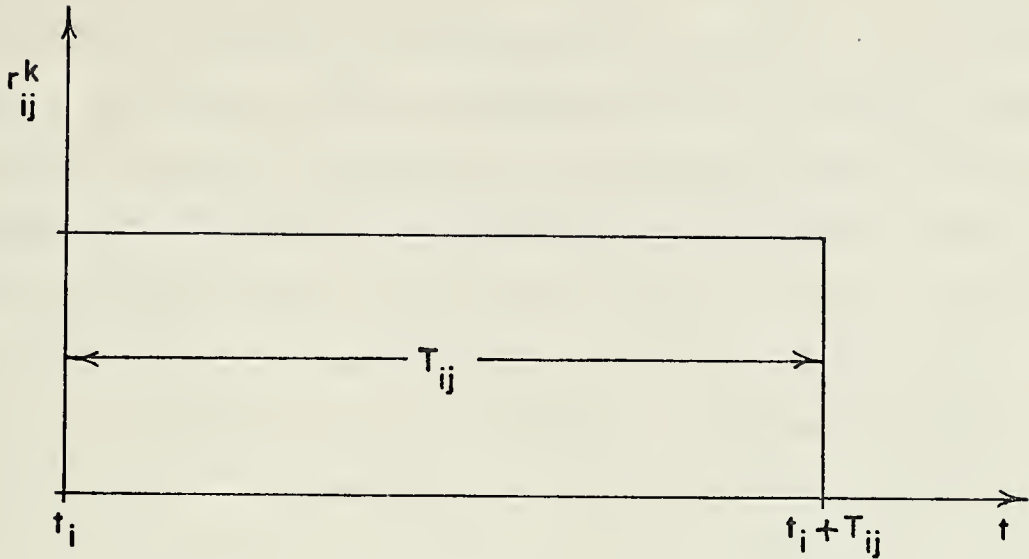
Figure 26

B. CONSTANT RESOURCE AVAILABILITY

1. The Network Job-Shop Scheduling Problem

The term network job-shop scheduling problem is a
name given, in this thesis only, to a scheduling problem
which has been approached by several authors [2], [3], [4],
[34], [67], [70], [71], [77], [78]. In each of these papers
the problem was called project scheduling subject to resource
constraints. The special name is used to show the distinc-
tion between this problem and the more general type of
problem developed in Chapters III and IV. The attributes of
the network job shop scheduling problem are first described
and then the relation between it and the more general
scheduling problem are outlined.

The network job-shop scheduling problem objective is to minimize the duration of a project composed of a number of activities connected by precedence relations in the usual manner. Associated with each activity is an activity duration and a resource necessary for performing the activity. Examples of resources in this context are a machine and a fixed size work crew. The resource cannot be split and can be assigned to only one activity at a time. This is the resource constraint for this problem. The fixed activity duration is the time required by the resource to process the activity from start to finish. A special case is the n job m machine job-shop scheduling problem. That is, given n jobs and m machines minimize the time to process all jobs on all machines where the order of machines and machine processing time for each job are given. A network representation of this type of problem was given in Figure 6 of Chapter II.

It is quickly apparent that if in the problem described in Chapters III and IV each shop is considered as a fixed resource then the problem can be viewed in the framework of the network job-shop problem. The assumptions necessary for a mixed integer programming representation for this case were given in Section III-D-2. So, if a constant number of workmen are available from each shop and if this fixed number of workmen is viewed as a work crew, the resource constrained project scheduling problem with fixed manday requirements becomes a network job-shop scheduling problem.

The shipyard scheduling problem can then be solved by
methods such as those suggested by the authors listed above.

2. Available Solution Techniques

A few of the most pertinent solution procedures that
can be used to solve network job-shop scheduling problems
and hence the shipyard problem with the additional assumptions
are now briefly described.

a. Implicit Enumeration

Because of the combinatorial nature of network
job-shop scheduling problems the most frequently used solu-
tion procedures are implicit enumeration or branch and bound
methods. A theory of implicit enumeration for combinatorial
problems has been examined by several authors [10], [60],
[78]. More specialized applications of branch and bound
methods have been constructed for the traveling salesman
problem [55], integer programming [1], [5], [32], [50],
scheduling [13], [41], [43], [58], [67], [76], [77] and, of
course, many other problem areas.

The use of implicit enumeration in solving net-
work job-shop scheduling problems is best exemplified by a
paper by Schrage [76]. In his paper a correspondence is
drawn between the possible permutations of the activities in
the project and all the active schedules. A schedule is an
assignment of starting times to all activities that does not
violate the resource constraints. A schedule is active if
no activity start times can be decreased without changing
the starting times of any other activities. It has been

shown [19], [44], [67] that in order to find an optimal
schedule, it is only necessary to search among the active
schedules. Schrage's paper gives a procedure for nonredun-
dantly enumerating all active schedules. The procedure is
valid for problems whose objective functions are non increasing
functions of the activity start times. This includes the
objective of minimizing project duration.

For the project duration minimization objective
it is not necessary with Schrage's procedure to explicitly
evaluate each active schedule. Two lower bounding methods
are given for implicitly evaluating each active schedule and
discarding those schedules which are not as good as some ex-
isting schedule. Some extensions of the method are also given
in the paper. These generalizations, when made, also apply
to the shipyard scheduling problem with appropriate changes
in the assumptions made.

b. Disjunctive Graphs

The representation of scheduling problems by dis-
junctive graphs has led to some interesting solution proced-
ures. The use of disjunctive graphs in scheduling has been
examined by Balas [2], [3], [4], Charlton and Death [15],
[16], Gorenstein [34], and Raimond [70], [71]. The method
of Balas [2] is briefly described here to illustrate the
role that disjunctive graphs play in resource constrained
project scheduling. A few definitions must first be given.

A disjunctive graph denoted $G = (X,A,B)$ is a
directed graph composed of a set X of nodes or events, a

set A of conjunctive arcs and a set B of disjunctive arcs.
A conjunctive arc is a directed arc in the usual sense. A
disjunctive arc (i,j) is a pair of arcs (i,j) and (j,i) con-
necting two nodes i and j. At most one of this pair of arcs
may be traversed by a path from the source to the sink of
the network. Associated with each disjunctive arc are dura-
tions d_{ij} and d_{ji}. A selection of a disjunctive arc implies
a specification of the arc's direction. If a complete selec-
tion or a specification of direction for all disjunctive arcs
in a network is made then the disjunctive graph is simply a
directed graph $G = (X,A)$. This graph may or may not contain
loops.

The problem confronted by Balas, Gorenstein, and
Raimond is called the network job-shop scheduling problem in
this thesis. That is, the objective is to minimize project
duration subject to precedence relations and a resource con-
straint. This constraint is that the resource may be employed
on only one activity at a time. Once again, the length of
time that the resource must be employed on each activity is
specified.

In the absence of resource constraints this
problem may be stated as a simple critical path problem

(V-B-1)   Minimize $t_n - t_1$

Subject to

(V-B-2) $t_j - t_i \geq d_{ij}$

t_i unrestricted.

132

In the network job-shop scheduling problem disjunctive arcs can be used to show the sharing of a common resource. The scheduling problem must first be represented by a network in which each activity has a unique starting time. That is, each activity to be scheduled must be the only activity incident from the node representing the commencement of that activity. This may require the addition of dummy nodes and arcs. The nodes representing the starting events for each activity sharing a common resource are then connected by disjunctive arcs. The placement of disjunctive arcs represents a sequencing of the activities through the constrained resource. The durations d_{ij} and d_{ji} of the disjunctive activity (i,j) are the same as the durations of the activities (i,k) and (j,p) respectively. That is, the durations of the single activities incident from nodes i and j.

This can be more readily understood by looking at an example. Suppose a project is described by the network in Figure 27(a). The pair of numbers on each activity represents first the resource type and then the duration of the activity. There are two resource types in the example. Each activity duration is the length of time necessary to accomplish a specified number of resource-time units using the specified resource. Note that activities $(1,2)$, $(1,4)$ and $(2,4)$, $(2,3)$ do not have unique starting events. It is then necessary to add artificial activities and nodes. This is illustrated in Figure 27(b). Ordinarily dummy activities would be represented by dashed lines but here they are not.

133
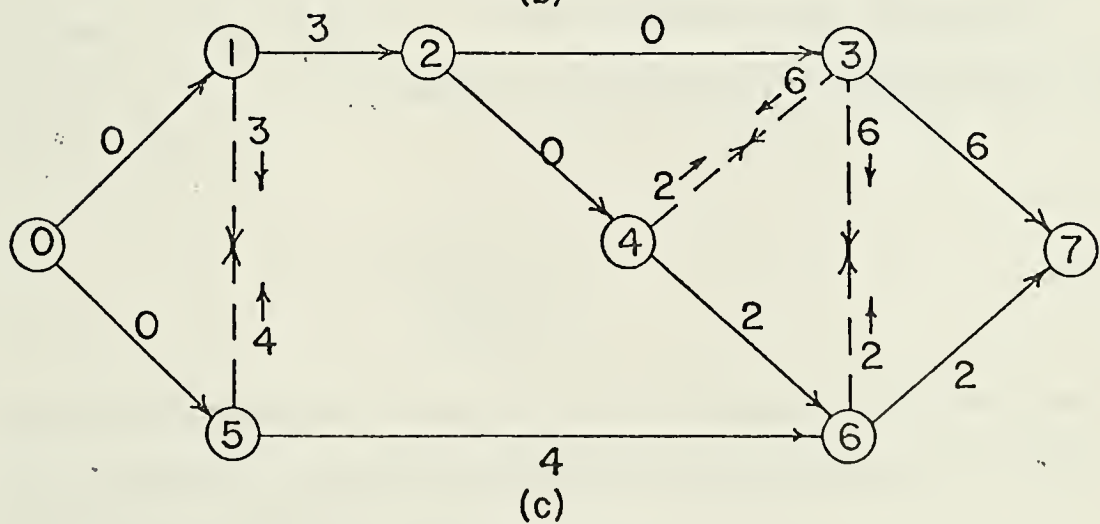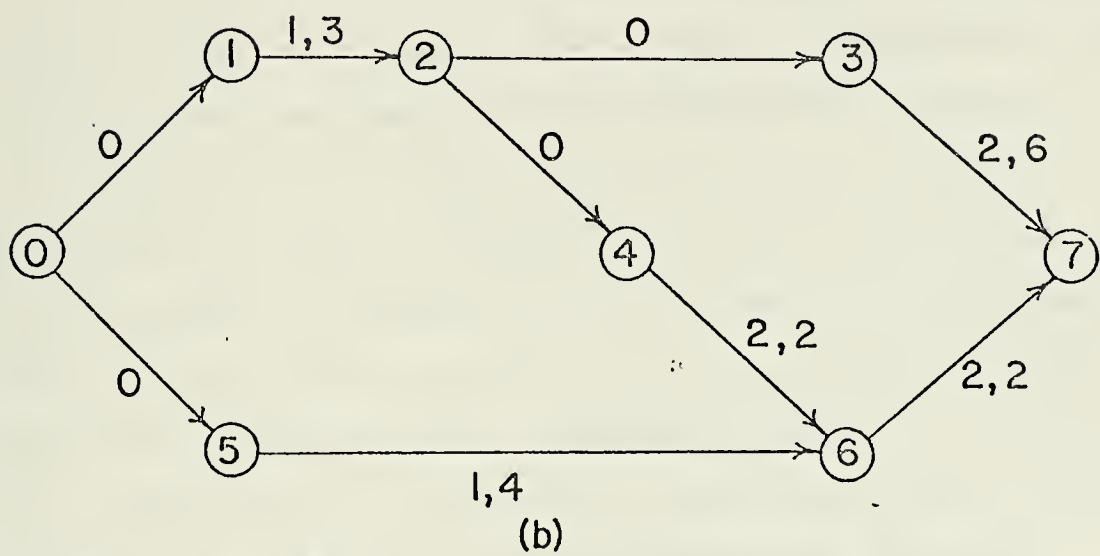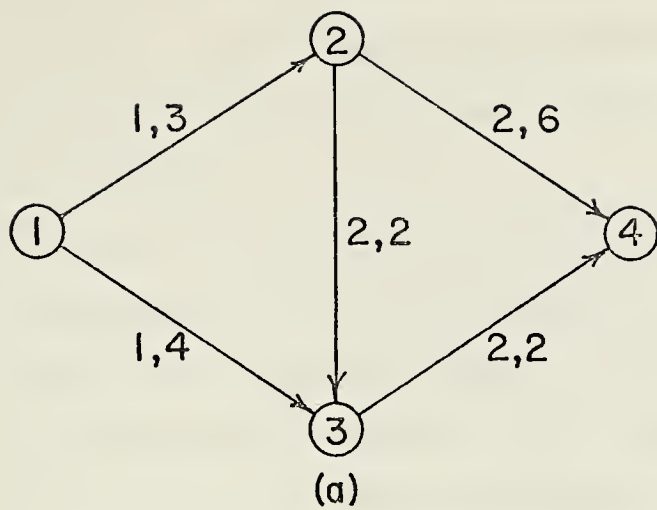
figure 27

This is to prevent confusion between dummy activities and disjunctive activities. The dummy activities have no resource associated with them. Figure 27(b) represents the same project as Figure 27(a).

In order to represent the constraints on the two available resources, disjunctive activities are added. To show the common use of resource type 1, nodes 1 and 5 are connected by the disjunctive activity (1,5). The same is done for nodes 3, 4, and 6 to show the sharing of the second resource. This is shown in Figure 27(c). The numbers on the activities are now simply activity durations. Note that $d_{15} = 3$ and $d_{51} = 4$. Since only one of the two possible orientations of (1,5) can apply, a fixing of a direction of (1,5) specifies a sequencing of (1,2) and (1,4) on resource type 1. Any (acyclic) selection for the network of Figure 27(c) then corresponds to a (feasible) schedule for the project. Associated with each acyclic selection G_k is a critical path with length v_k. If V is the set of all acyclic selections G_k of the disjunctive graph G, a minimaximal path in G is given by

$$v^* = \min_{G_k \varepsilon V} v_k.$$

The minimum project duration with restrictions on the resources corresponds to the selection of a minimaximal path in the disjunctive graph G.

The methods for finding a minimaximal path in G vary. Balas [2], [3], and Raimond [70] formulated the problem as a mixed integer programming problem. Balas applied the Benders decomposition procedure [7] using his additive algorithm to solve the associated zero-one integer programming problem. A sequence of critical path problems generates the coefficients of the zero-one variables in the constraints that are added at each iteration. Raimond's method consisted of an application of mixed continuous-zero-one programming by direct search developed by Lemke and Spielberg [53]. The number of zero-one variables in each case is equal to the number of disjunctive activities in the disjunctive graph. A zero-one variable y_{ij} associated with a disjunctive activity (i,j) is zero if the activity is oriented in one direction and one if the activity is complemented (oriented in the opposite direction).

Balas [4] developed an implicit enumeration method for finding a minimaximal path in a disjunctive graph which is superior to his previous efforts. No integer programming problem need be solved. A sequence of critical path problems is generated and only disjunctive activities on the critical path of the previous selection are complemented. No computational experience is given, so it is difficult to ascertain how large a problem may be solved using the method.

More recently, Samuel Gorenstein [34] modified some of the earlier methods to more quickly solve the network job-shop scheduling problem. His method accommodates the

136

presence of more than one of each resource type. The algorithm
presented uses partial enumeration to solve a mixed integer
programming problem. The largest problem solved optimally
by Gorenstein was a five job eight machine job-shop scheduling
problem with 3 of each machine type.

These methods have been briefly described to show
what methods are available for solving the shipyard scheduling
problem when a fixed crew size is available for employment on
an activity. Methods used to approach the case where there
is more than one of each resource type available [3], [34]
can also be used in the shipyard problem. This can be done
by simply assuming there is more than one crew of fixed size
available from each shop. An example of the formulation and
solution of a project using disjunctive graphs is given next.

c. <u>An Example</u>

The duration of the project represented by the
network of Figure 27 can be minimized using disjunctive
graphs. The representation by disjunctive graph is given
by Figure 27(c). The relations (V-B-1), (V-B-2) for a simple
CPM problem can be extended to model a disjunctive graph.
The problem becomes

(V-B-3)   Minimize $t_n - t_1$

Subject to

(V-B-4) $t_j - t_i \geq d_{ij}$; $\forall (i,j) \epsilon A$

(V-B-5) $\qquad t_p - t_k \geq d_{kp}$

$\qquad\qquad\qquad\qquad\qquad \forall (i,j) \epsilon B.$

(V-B-6) $\qquad$ or $t_k - t_p \geq d_{pk}$

$\qquad\qquad\qquad t_i$ unrestricted.

In this formulation, A is the set of conjunctive activities
and B is the set of disjunctive activities. Disjunctive
constraints of the form (V-B-5), (V-B-6) have been dealt
with by many authors. Balas and Raimond formulated the prob-
lem in a manner similar to that below. If a zero-one variable
y_{ij} is associated with each disjunctive activity and if
$y_{ij} = 0$ signifies orientation in one direction and $y_{ij} = 1$
signifies the opposite orientation, the problem (V-B-3)-(V-B-6)
can be stated as a mixed integer programming problem with
zero-one variables. This formulation is

(V-B-7) $\qquad\qquad$ Minimize $t_n - t_1$

Subject to

(V-B-8) $\qquad t_j - t_i \geq d_{ij} \; ; \; \forall (i,j) \epsilon A$

(V-B-9) $\qquad t_p - t_k + My_{kp} \geq d_{kp}$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad (k,P) \epsilon B$

(V-B-10) $\qquad t_k - t_p - My_{kp} \geq d_{pk} - M$

$\qquad\qquad t_i$ unrestricted, $y_{ij} = 0$ or 1, where M
is some sufficiently large positive number.

Following the procedure of Benders, for fixed y_{ij} (i.e., a selection of disjunctive activities), the problem becomes a critical path problem with some redundant constraints. The dual of (V-B-7)-(V-B-10) for fixed y is

(V-B-11)   Maximize $\sum\limits_{(i,j)\epsilon N} d_{ij}u_{ij}$

Subject to

(V-B-12) $Eu = \begin{bmatrix} -1 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ 1 \end{bmatrix}$

$u_{ij} \geq 0$

where E is the associated node-arc-incidence matrix. Also $d_{ij}, \forall(i,j)\epsilon N$ is the right-hand side of (V-B-8)-(V-B-10) with terms involving M and y transposed. This new right-hand side is represented by (d - Dy). The maximum of (V-B-11) occurs at an extreme point of (V-B-12). For fixed y, a critical path solution to (V-B-7)-(V-B-10) yields an extreme point of (V-B-12). That is, if (i,j) is on the critical path, $u_{ij} = 1$ and if (i,j) is not on the critical path, $u_{ij} = 0$. At each iteration a zero-one program of the form given below must be solved.

(V-B-13)   Minimize z

Subject to

(V-B-14) $z \geq (d - Dy)u$

                       $y = 0$ or 1.

One additional constraint is added at each iteration and each
constraint is of the form

$$z \geq \sum_{\substack{(i,j) \\ \varepsilon(A \cap X)}} d_{ij} + \sum_{\substack{(i,j) \\ \varepsilon(B^+ \cap X)}} (d_{ij} - My_{ij}) + \sum_{\substack{(i,j) \\ \varepsilon(B^- \cap X)}} (d_{ij} - M + My_{ij})$$

where A is the set of conjunctive arcs, B^+ is the set of
"normally" oriented disjunctive arcs and B^- is its complement.
X is the set of all activities for which $u_{ij} = 1$. Normally
oriented refers to the direction established for the activity
(i,j) when $y_{ij} = 0$. This is simply established by convention.
The procedure terminates when $z = v^k$ where v^k is the length
of the critical path at the Kth iteration.

          ·The example is now continued. Table IX shows the
coefficients for the mixed integer program representing
Figure 27(c). Start the procedure by putting all $y_{ij} = 0$.
(This establishes the normal orientation for each disjunctive
activity.) Then find the critical path of the directed net-
work generated. The critical path is shown in Figure 28 as
a heavy line. The critical path length is 13 and the values
of u_{ij} are $u_{01}=u_{12}=u_{23}=u_{34}=u_{46}=u_{67}=1$ and $u_{ij} = 0$ otherwise.

Table IX

t_1	t_2	t_3	t_4	t_5	t_6	t_7	y_{15}	y_{34}	y_{36}		d	u
-1	+1									$\geq$	3	u_{12}
		-1				+1				$\geq$	6	u_{37}
			-1	+1						$\geq$	2	u_{46}
				-1	+1					$\geq$	4	u_{56}
					-1	+1				$\geq$	2	u_{67}
-1				+1			+M			$\geq$	3	u_{15}
+1				-1			-M			$\geq$	4-M	u_{51}
		-1	+1					+M		$\geq$	6	u_{34}
		+1	-1					-M		$\geq$	2-M	u_{43}
		-1			+1				+M	$\geq$	6	u_{36}
		+1			-1				-M	$\geq$	2-M	u_{63}
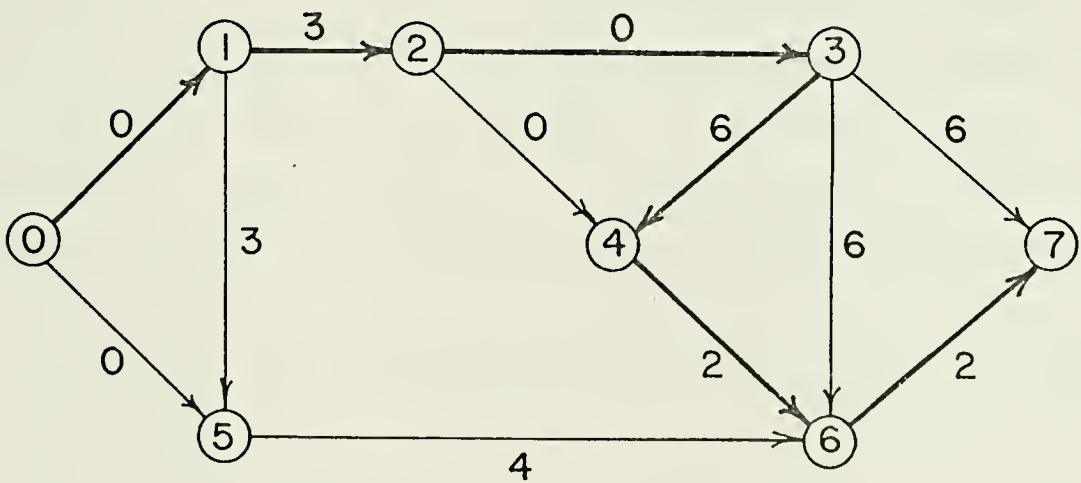
Figure 28

The first constraint generated for the zero-one program is then

$$z \geq 13 - My_{34}.$$

The u_{ij}'s which minimize z in this case are $y_{34} = 1$. All other y_{ij}'s are maintained at zero and the new critical path problem is solved. The procedure stops when $z = v^k = 13$. Table X shows the progress of the iterations and Figure 29 is the network representing the optimal solution.

Table X

k	d	y_{15}	y_{34}	y_{36}	Solution	z	v^k
1	13		-M		$y_{34}=1$	13-M	13
2	13-M		+M	-M	$y_{36}=y_{34}=1$	13-M	13
3	15-M	-M		+M	$y_{15}=y_{36}=y_{34}=1$	15-M	17
4	17-2M	+M		+M	$y_{15}=1$	13	17
5	17-M	+M	-M		all $y_{ij}=0$	13	13
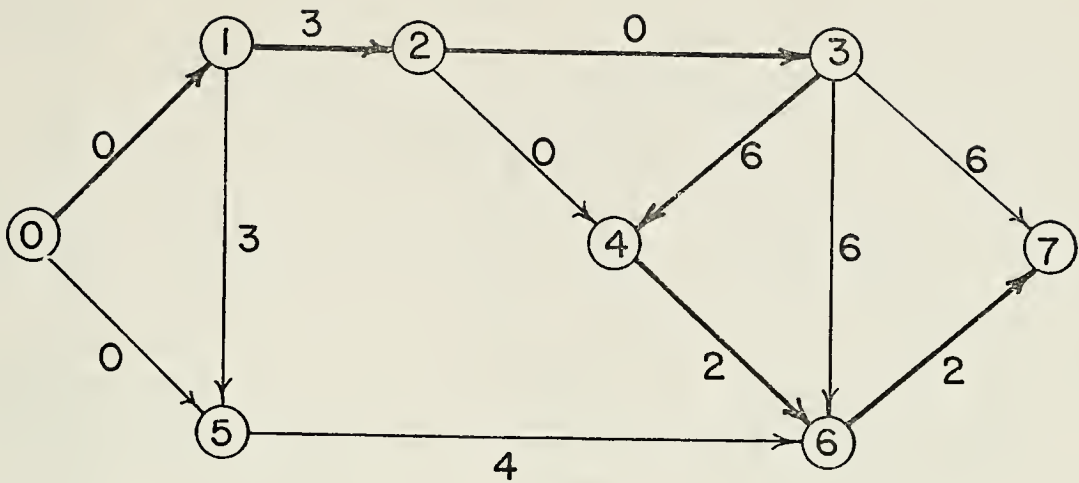
Figure 29

C. VARIABLE RESOURCE AVAILABILITY

The heading for this section means that there is a re-
source availability profile associated with each shipyard
shop. A typical resource profile is shown in Figure 30.
The difference between a resource availability profile in
this section and the profile of Chapter IV is that the re-
source availability breakpoints occur at fixed points in
time, not at the variable event times. For any point in
time T*, the resource availability profile gives the number
of men available R^k_{T*} . The existence of the variable resource
availability profile makes the scheduler's job increasingly
difficult. The only successful approaches to the minimiza-
tion of duration in a resource constrained project with a
variable resource availability profile have been heuristic

143
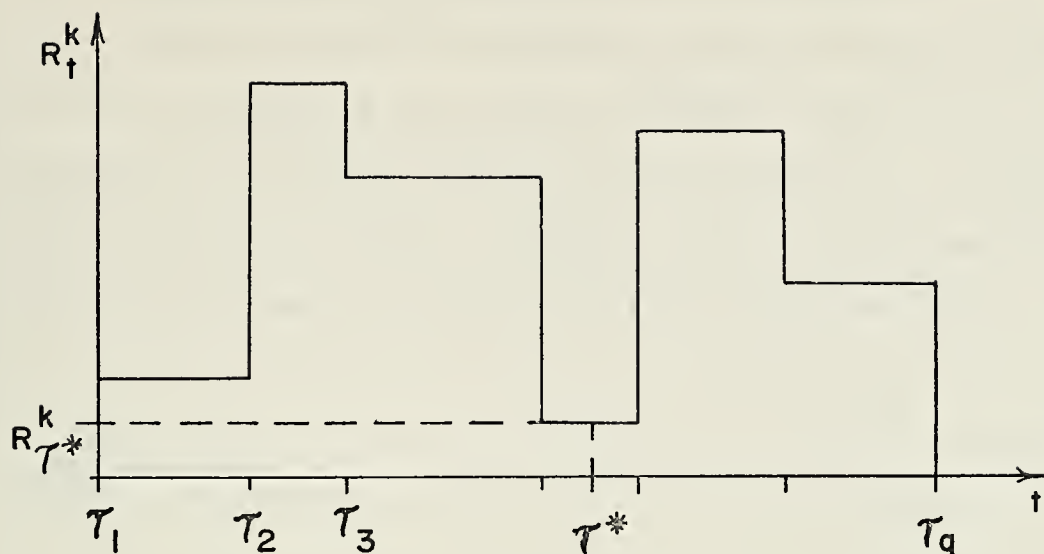
Figure 30

in nature. Only one approach, that of Karush [44], has
dealt directly with the problem analytically. Chapter III
of this thesis provides a mixed integer programming formula-
tion for this problem. Jerome Wiest [83], [84], [85] and
Moder and Phillips [61, p. 158] have applied heuristic rules
to get a feasible and hopefully a good solution to the problem.

Any successful approaches in this area would be quite
meaningful for the shipyard scheduler. The resource avail-
ability profile is identical to the form of availability of
shipyard shop workers. Once again, activity resource usage
profiles are required to be specified in advance.

1. Nonincreasing Activity Resource Profiles

For the case where activity resource usage profiles are nonincreasing, Karush [44] developed a method for enumerating the schedules necessary for comparison to obtain minimum project duration. Examples of nonincreasing activity resource profiles are shown in Figure 31(a) and (b).

(a)

(b)
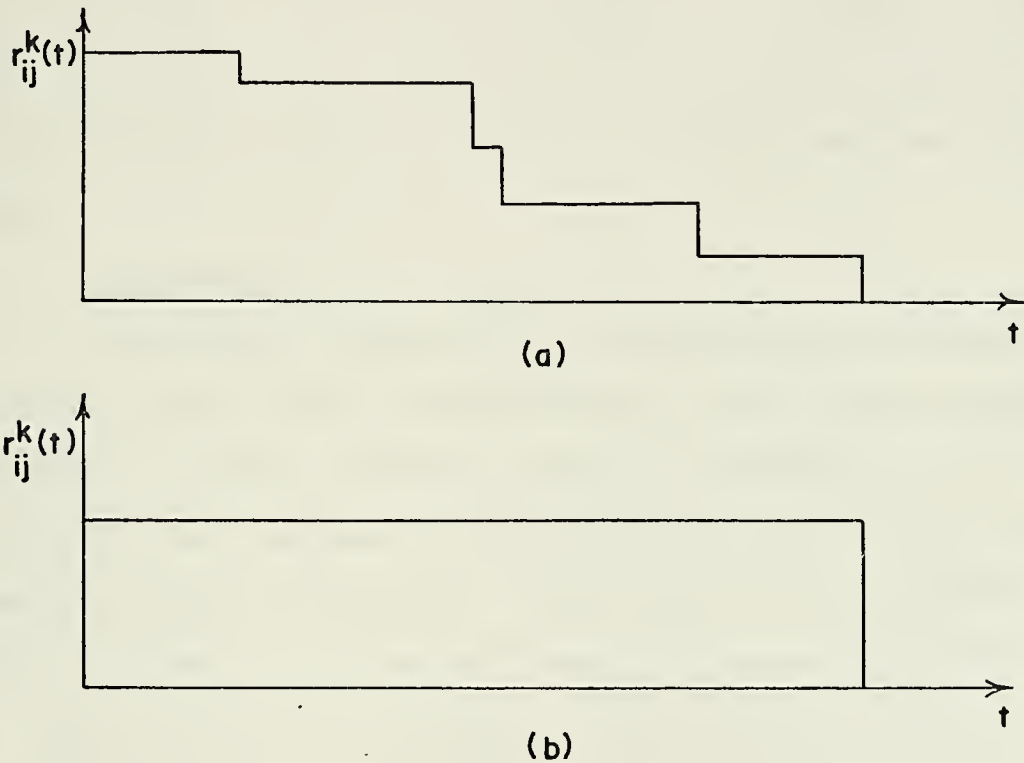
Figure 31

Karush called the set of schedules that require evaluation the set of all left-packed schedules. This term is very similar to the term "active schedule" defined earlier but is more appropriate in the context of a variable resource availability profile.

In order to obtain all of the left-packed schedules, it is necessary to generate all feasible sequences of activities and schedule each of these activities as early as possible in the order they appear in the sequence. If each activity in the project has associated with it a nonincreasing resource profile, a minimum duration schedule will be found by evaluating the durations of all the schedules generated in this manner. Since it is necessary to completely enumerate all left-packed schedules, this procedure presents much more difficulty than the constant activity resource profile case.

2. An Example

The use of sequential left-packing as described by Karush is most easily understood by looking at an example. Consider the small project network of Figure 32(a). The activities are numbered arbitrarily 1 through 5 and these numbers appear on the arcs. The resource usage profile for each of these activities are shown in Figure 32(b)-(f). It is desired that the activities be scheduled so that the project duration is a minimum and the available resources are not exceeded. The resource availabilities are given by the project resource profile of Figure 33. All feasible permutations of the activities are then generated. A permutation is infeasible if the precedence relations are violated. For each of these permutations the activities are scheduled as early as possible. That is, if activity i must immediately precede activity j because of the form of the project network,

146
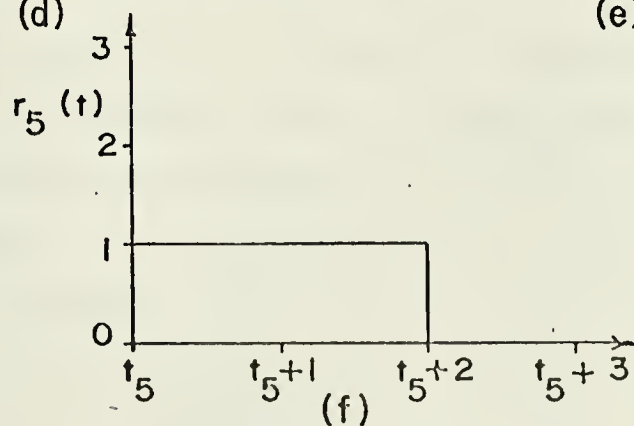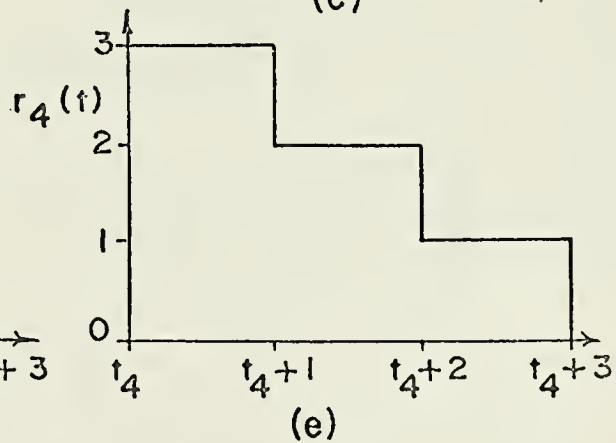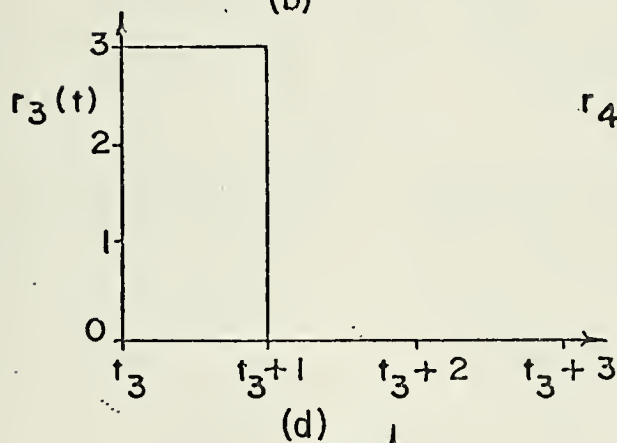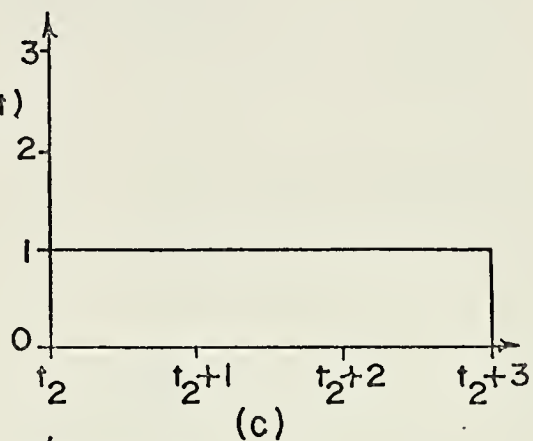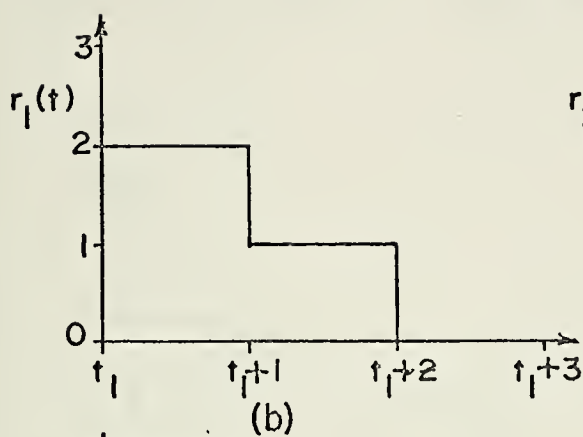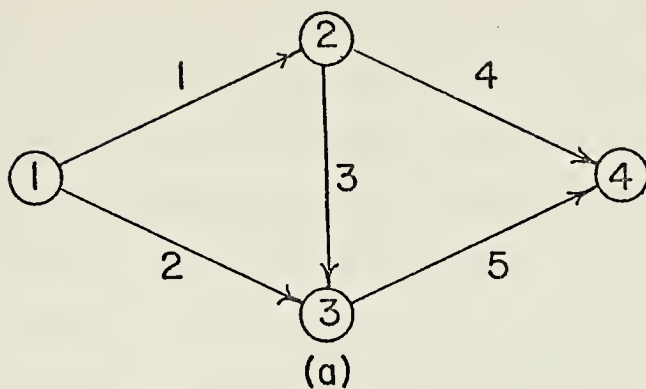
figure 32

then activity j may not be commenced until activity i has
been completed. If, however, activity k does not immediately
precede activity j in the network but does precede j in the
permutation, then activity j may be scheduled at the same
time as activity k providing sufficient resources are
available. The start time of activity j in this case must
not be less than the start time of activity k.
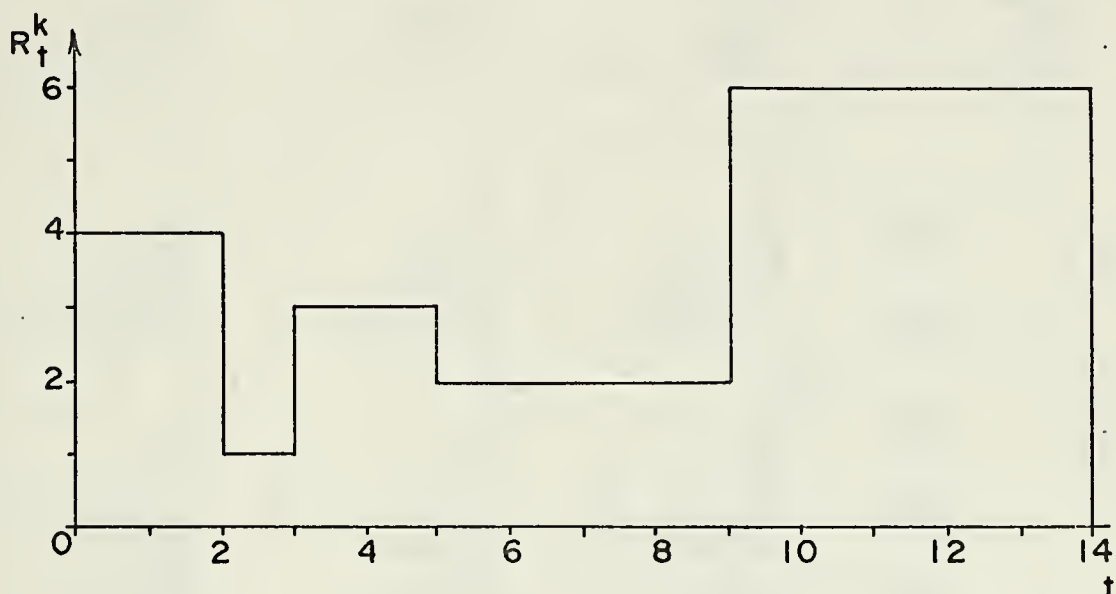
Figure 33

For the example, the possible permutations and the
associated schedules are shown in Figure 34(a)-(k). There
is a tie between the schedules (a) and (d) for the minimum
duration. Each of these schedules can be completed in eight
days and are optimal.

figure 34

149
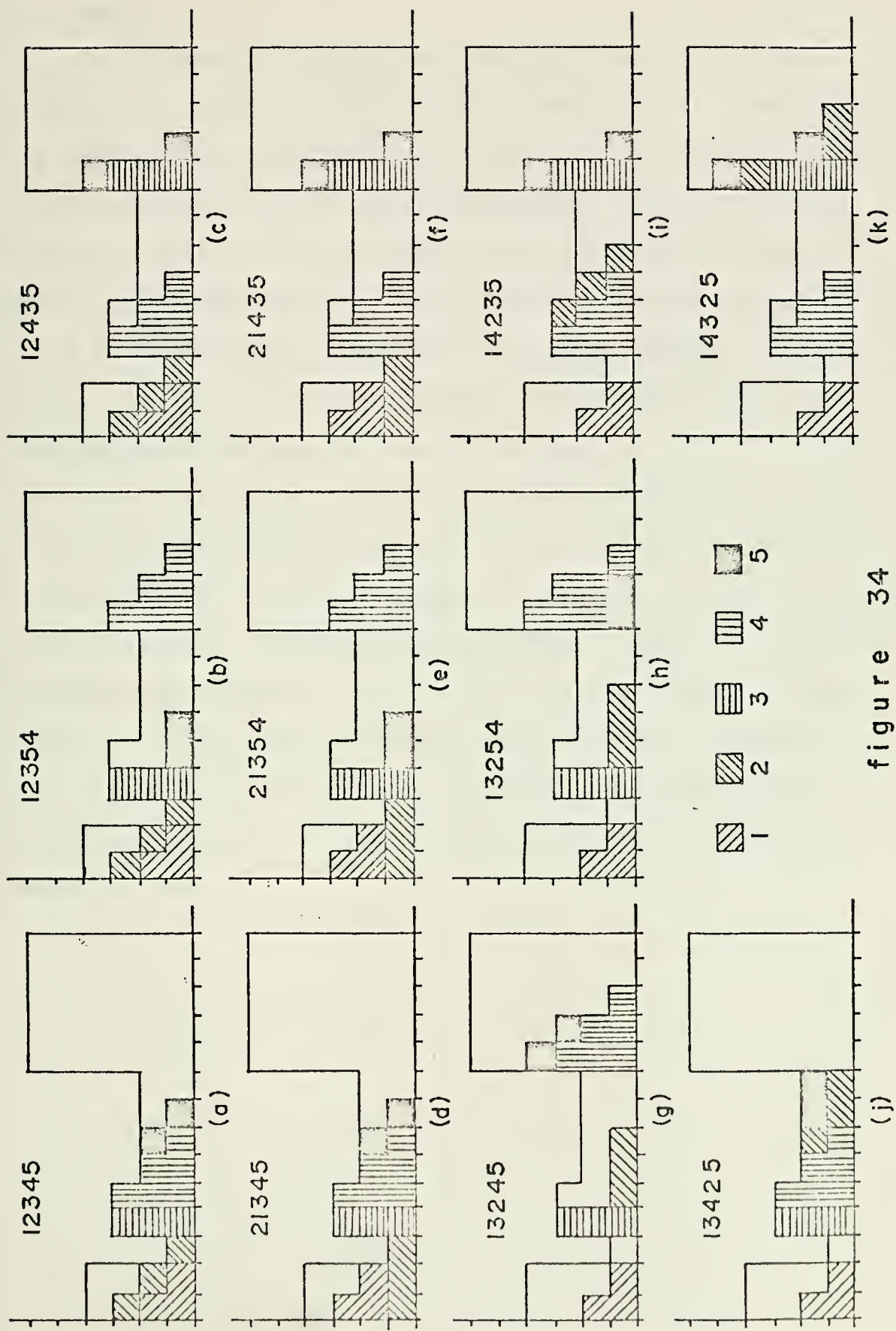
D. SUMMARY

If a manager is willing to make certain simplifying
assumptions concerning the method of employment of resources
the problem of minimizing project duration subject to resource
constraints and fixed manday requirements can be placed in
a form for which solution methods are presently available.
Some of these assumptions can be quite realistic and in many
cases do not seriously weaken a solution obtained. For in-
stance, the fixed crew size assumption is often a very
reasonable assumption for many projects.

A difficulty, once again, is in the combinatorial nature
of scheduling problems. Even with the simplifying assumptions,
it is difficult to use existing methods to solve realistically
large problems. Wiest's heuristic SPAR-1 model and the Moder
and Phillips heuristic [61, p. 158] still provide the best
methods for actually scheduling large projects. Although
they cannot guarantee optimal solutions, they can yield
feasible schedules for very large projects in a reasonable
amount of time.

VI.   DIRECTIONS FOR FURTHER RESEARCH

Investigation into a problem as complex as shipyard sched-
uling is certain to lead to many unanswered questions. The
shipyard scheduling problem has a very general structure which
encompasses many smaller scheduling problems. There is a
great deal of work to be done both on the smaller problems
and the general scheduling model. Much of the progress on
the smaller problems such as the job shop scheduling problem
will depend on advances in combinatorial methods. As these
procedures improve the special cases of the shipyard problem
will be solved more easily also.

The mixed integer programming model proposed in Chapter
III gives a very meaningful description of the total cost
problem. Additionally, this model provides a structure for
typing together several scheduling problems which are solvable.
As a computational technique, however, the model is not directly
applicable. Geoffrion and Marsten [33] have described some
research that is attempting to improve the computational
ability of Benders' algorithm. At present, most of these
methods involve approximate solutions to the zero-one problem
such as obtained through the use of continuous linear pro-
gramming up to some specified point in the iterations. Ad-
vances in the computational efficiency of this powerful pro-
cedure will most likely allow the mixed integer programming
model to be useful for modest size projects. In order to
utilize any new methods it is still necessary to find an

initial basic feasible solution to the dual of the partitioned total cost problem (M2D). This is not a trivial task and the discovery of a method for finding this would definitely be worthwhile.

The transportation problem structure of the total cost problem is very interesting. It would certainly be gratifying to be able to utilize this structure directly. Perhaps the special structure of the zero-one problem could be used to generate costs for sequences of transportation problems thus allowing the use of efficient transportation algorithms. The mixed integer programming model would then be an outstanding model of the scheduling system and could even be adapted to solve the lesser included problems.

After the discovery of techniques like those described above, the scheduling model could be generalized even more. The introduction of minimum and maximum crew sizes would lead to a capacitated transportation problem structure. This would, in turn, require more research into the efficient solution of this type of transportation problem.

The nonlinear programming model of Chapter IV, like the mixed integer model, provides a good description of the system. This model has an efficient solution procedure for a special case of the shipyard total cost problem. There are several improvements that could be made on this model. The first and most desirable change would be the expression of the model in a form not requiring the strict event ordering assumption. This would improve the acceptability of any

152

subsequent solution procedures. Failing this, it would be worthwhile to develop an alternative to the complete enumeration of all possible event orderings. It is possible that some means of implicitly generating all orderings could be devised. A thorough testing of heuristic methods of event ordering might also prove fruitful.

A procedure for direct solution of the multiple resource nonlinear programming model would definitely be welcomed. This looks doubtful but any results in this area would warrant further investigation. Further examination of this problem using dynamic programming for nonserial systems [64, p. 184] might also be worthwhile.

Another area that could be more fully explored is that of making the variable and fixed resource profiles more closely coincident. A possible way to approach this would be to develop heuristic rules for separating activities. Along with this it would be necessary to assign the proper resource availability values between events. This can be done by trying every possible assignment of resources to inter-event intervals but more efficient methods would be better.

Because of the interconnection between the shipyard scheduling problem and many other scheduling problem types, advances in the methods of solving one will benefit the others. A worthwhile endeavor, then, would be to attempt to find solutions to problems like the job-shop scheduling problem with total cost objective functions. As the ability to solve large

combinatorial problems increases, the solution of these

complex scheduling problems will become easier.

VII. CONCLUSIONS

The shipyard cost minimization problem is sufficiently general to bring together many other scheduling problems. The job-shop and flow-shop scheduling problems, the network job-shop problem, and several other resource allocation problems can all be viewed as subproblems of the shipyard system. The formulations presented in this thesis can therefore describe many other scheduling environments. This structure is more general than that of most problems found in the literature of scheduling and, in fact, includes many of these scheduling systems.

The cost minimization objective function provides for more generality than the usual duration minimization criterion. If the proper parameter specifications are made, in fact, cost minimization becomes duration minimization. Additionally, the use of manday requirements to characterize activity completion is more general than the more common fixed crew size, fixed duration specifications. This again is a generalization since the fixed manday requirements model includes the fixed crewsize, fixed duration model after appropriate assumptions are made.

This thesis, in approaching the general shipyard problem, shows the relationship between it and several other problems. The relationship is strong enough to permit solving special cases of the shipyard scheduling problem with existing solution procedures. The limitations on the size of shipyard

project that can be dealt with depends on the efficiency of the existing method used. The operation of most of the present scheduling methods is hindered by the combinatorial nature of even the smallest scheduling problem.

The mixed integer programming and nonlinear programming formulations of this complex problem allow more complete understanding of the shipyard total cost problem as well as the included sub problems. The mixed integer model represents the total cost problem as several transportation problems linked by precedence constraints. Advances in integer programming solution procedures will improve the acceptability of this mixed integer model. Fixing the event times of the nonlinear cost model led to several generalized transportation problems which were not connected by precedence relations. The mixed integer programming model with fixed event times is, in fact, a special case of this nonlinear programming model. The application of dynamic programming to the single resource case of the nonlinear programming model led to an efficient solution procedure. This method finds a minimum cost solution for a large project network very quickly.

A. <u>SUMMARY CLASSIFICATION OF SCHEDULING PROCEDURES</u>

This appendix references several papers concerning resource constrained project scheduling in the format of Chapter II. An Arabic numeral represents a reference in the bibliography while a Roman numeral and letters in parentheses represent a section in this thesis.

A. OPTIMIZATION CRITERIA

    1. <u>Resource Allocation</u>

        a. Cost Minimization: 20, 38, 58, 84, (III-E), (IV-E).

        b. Duration Minimization: 2, 3, 4, 9, 13, 15, 16, 21, 26, 34, 35, 41, 43, 44, 47, 49, 56, 57, 67, 69, 70, 71, 73, 74, 76, 77, 82, 83, 84, 85, (III-C), (IV-D), (V).

    2. <u>Resource Leveling</u>: 14, 49, 54, 62, 68, 72, 73, 81.

    3. <u>Time/Cost Tradeoff</u>: 17, 18, 27, 30, 42, 45, 46, 48, 65, 66, 79.

B. NATURE OF RESOURCE CONSTRAINTS

    1. <u>Resource Availability Profiles</u>

        a. Fixed Resource Availability: 38, 44, 47, 49, 67, 84, (III), (V-C).

        b. Variable Resource Availability: 65, (IV).

        c. Constant Resource Availability: 2, 3, 4, 11, 13, 15, 16, 21, 26, 34, 35, 41, 43, 57, 67, 69, 70, 71, 76, 77, (V-B).

    2. <u>Bounded Resource Volume</u>: 9, 73, 74.

C. ACTIVITY CHARACTERISTICS

    1. <u>Activity Duration Estimate</u>: Same as for B-1-c.

    2. <u>Fixed Resource-Time Unit Requirements</u>: 9, 20, 68, 72, 73, 74, 81, (III), (IV).

    3. <u>Resource Usage Profiles</u>: 44, (V-C).

B.   COMPUTATIONAL RESULTS

Algorithm 3 of Section IV-E-2 is a method for minimizing project cost with a single constrained resource.   The FORTRAM IV computer program immediately following this appendix was run on the Naval Postgraduate School's IBM 360/67 computer to test the algorithm.   The results of these tests are presented in Table XI and Figure 35.

Table XI

Problem Number	Number of Events	Number of Activities	Execute Time (Sec.)	Total CPU Time (Sec.)	Storage (k-bytes)	Remarks
1-9	10	13	1.30	6.87	54	1
10-25	10	13	1.96	7.74	54	1
26-35	10	13	1.69	7.92	54	1
36-57	10	13	-	7.62	54	1,2
58-70	10	13	-	6.77	54	1,2
71	10	13	0.53	6.16	54	3,7
72	34	44	0.71	6.89	56	4,7
73	58	75	0.99	6.47	56	5,7
74	82	107	1.33	7.23	56	6,7
75	106	138	1.59	7.33	56	6,7
76	130	169	1.92	7.75	56	6,7
77	154	200	2.43	9.00	56	6,7

Remarks

1. The 70 possible event combinations for the example of Section IV-E-2.

2. Used UCLA's QUICKRUN which gives only total CPU time.

3. The example of Section IV-E-2.

4. Network only [59, p. 73].

5. Network only [28, p. 99].

6. Network and parameters generated randomly.

7. Plotted in Figure 35.

A total of 77 problems were solved using the cost mini-mization program. In order to provide some continuity among the projects tested, only projects with the same network density were used. Johnson [43] defined the network density to be the ratio of the number of events to the number of activities. All of the networks of Table XI have a density of 0.77. Each subsequent project after problem 71 has 24 more events than the preceeding project.

The first 70 problems consist of all the feasible event orderings for the example in Section IV-E-2. These were generated by hand using the procedure suggested in Section IV-E-1. Problem 71 is the example of Section IV-E-2 and is one of the feasible orderings of problems 1-70. This project was run alone where the first 70 problems were tested in the groups indicated in Table XI. The minimum total project cost for problems 1 through 70 ranged from $2378 to $3825. Only one feasible ordering yielded the minimum cost and this was problem 54 illustrated in Figure 19. In this case the ordering was the one generated by the heuristic rule of IV-F-1. This will not be true in every case, however.

Problem 72 uses only the network of a factory extension problem of McLaren and Buesnell [59, p. 73]. The costs and manday requirements were contrived by this author. The network for problem 73 was a modified version of a nuclear reactor construction project [28, p. 99]. Problems 74-77

160

figure 35

161
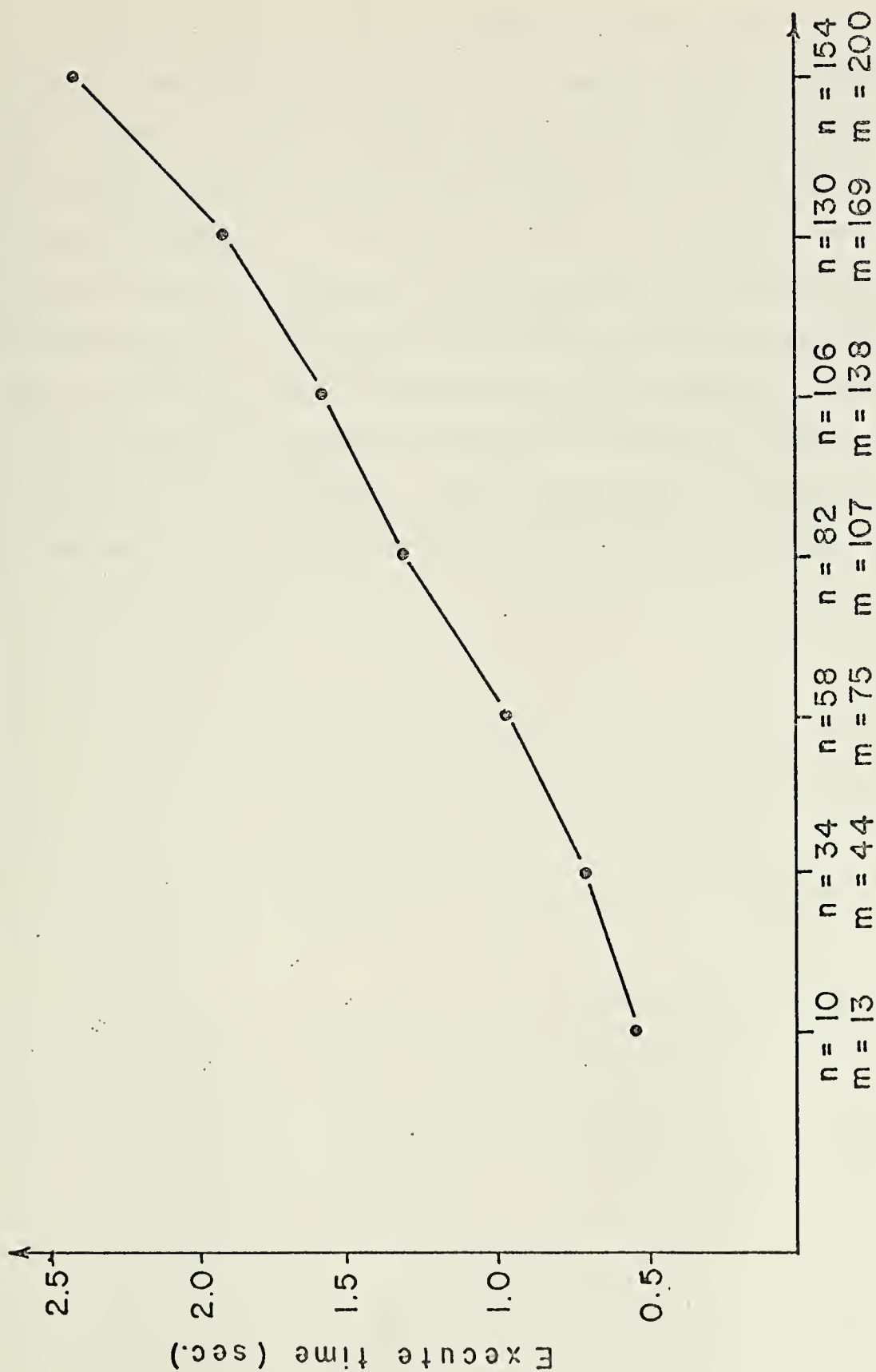
were developed in a random fashion, ensuring that network density was constant at 0.77. The largest project attempted was number 77 with 154 events and 200 activities. The execution of this problem only required 56×10^3 bytes of computer storage. In order to prepare this project for solution by the program, it was necessary to key punch 361 data cards. The program requires one card for each event and activity and seven parameters must be entered. The instructions for entering this data are given at the beginning of the program. The computational results for problems 71-77 are plotted on the graph of Figure 35.

```
C    ****************************************************************
C                    COST MINIMIZATION IN PROJECT NETWORKS
C
C       THIS PROGRAM DETERMINES THE NORMAL AND OVERTIME RE-
C       SOURCE ALLOCATIONS THAT WILL MINIMIZE TOTAL PROJECT
C       COST.   TOTAL PROJECT COST CONSISTS OF NORMAL MANDAY
C       COST, OVERTIME COST AND A PENALTY COST FOR EXCEEDING
C       A TARGET DATE.  A SINGLE RESOURCE IS CONSTRAINED BE-
C       TWEEN THE EVENTS.  A FIXED NUMBER OF MANDAYS MUST BE
C       EXPENDED TO COMPLETE AN ACTIVITY.
C
C       SUBROUTINES CALLED:
C
C           DURMIN:   CALCULATES THE MINIMUM DURATION FOR THE
C                     EXISTING COMBINATION OF XIJ (NORMAL MAN-
C                     DAYS) AND (OVERTIME MANDAYS).
C
C           COSTMN:   COSTMN SOLVES THE LINEAR PROGRAMMING PROB-
C                     LEM OF STEP 3 OF THE ALGORITHM AND YIELDS
C                     OPTIMAL OVERTIME ALLOCATION.
C
C           REDUOT:   PERFORMS THE PARAMETRIC ANALYSIS ON OVER-
C                     TIME TO EXTEND PROJECT DURATION UP TO THE
C                     DUE DATE.
C
C           OUTPUT:   PRINTS OUT THE OPTIMAL ALLOCATIONS, TOTAL
C                     COST, RESOURCE AVAILABILITIES AND THE
C                     EVENT TIMES.
C
C       EVENTS ARE READ INTO THE PROGRAM BY LISTING THEM IN
C       THE FOLLOWING MANNER:
C
C       J, RAVAIL(J), J=1,2,...,NEVENT
C
C       WHERE:
C
C       J=EVENT NUMBER, RAVAIL(J)=RESOURCES AVAILABLE AT EVENT
C       J.  NEVENT IS THE NUMBER OF EVENTS IN THE PROJECT.
C       J IS AN INTEGER, RIGHT-JUSTIFIED IN COLUMNS 1-10, AND
C       RAVAIL(J) IS A FLOATING-POINT NUMBER WITH TWO DECIMAL
C       PLACES, RIGHT-JUSTIFIED IN COLUMNS 11-20.
C
C       ACTIVITIES ARE READ INTO THE PROGRAM BY LISTING THEM
C       IN THE FOLLOWING MANNER:
C
C       I, ISTART(I), IEND(I), COST(I), OTCOST(I), MANDAY(I),
C       I=1,2,...,NOACTS
C
C       WHERE:
C
C       I=ACTIVITY NUMBER, AN INTEGER RIGHT-JUSTIFIED IN COL-
C       UMNS 1-10.  ISTART(I)=INITIAL EVENT FOR ACTIVITY I,
C       AN INTEGER RIGHT-JUSTIFIED IN COLUMNS 11-20.  IEND(I)=
C       TERMINAL EVENT FOR ACTIVITY I, INTEGER, COLUMNS 21-30.
C       COST(I)=NORMAL COST PER MANDAY OF ACTIVITY I,
C       FLOATING-POINT WITH TWO DECIMAL PLACES IN COLUMNS 31-
C       40.  OTCOST(I)=OVERTIME COST PER MANDAY OF ACTIVITY I,
C       TWO-PLACE FLOATING-POINT, COLUMNS 41-50.  MANDAY(I)=
C       REQUIRED MANDAYS FOR ACTIVITY I'S COMPLETION, INTEGER,
C       COLUMNS 51-60.  NOACTS IS THE NUMBER OF ACTIVITIES IN
C       THE PROJECT.
C
C       THE NUMBER OF EVENTS (NEVENT), THE NUMBER OF ACTIV-
C       ITIES (NOACTS), THE DUE DATE (TARGET), THE LIMIT ON
C       OVERTIME (BOUND) AND THE PENALTY COST (PCOST) MUST BE
C       ENTERED IN THE MAIN PROGRAM BY THE USER.
C
C       LOW AND NOPROB ARE THE FIRST AND LAST PROBLEM NUMBERS
C       FOR MULTIPLE PROBLEMS AND ARE THE SAME IF A SINGLE
C       PROBLEM IS ENTERED.
C
```

```
C          THE COMMON STATEMENTS IN THE MAIN PROGRAM AND ALL SUB-
C          ROUTINES WILL ACCOMODATE UP TO 200 EVENTS AND 200
C          ACTIVITIES.  THESE MUST BE CHANGED FOR LARGER PROJECTS
C          THE DIMENSION STATEMENTS IN COSTMN AND REDLCT MUST
C          ALSO BE CHANGED IN THAT CASE.
C
C          ***************************************************
C
           CCMMON ISTART(200),IEND(200),COST(200),OTCCST(200),
          1MANDAY(200),RAVAIL(200),XIJ(200),SIJ(200),IALFA(200),
          2TIME(200),XAVAIL(200)
           NOACTS=200
           TARGET=500.00
           BCUND=200.00
           PCOST=3500.00
           LCW=76
           NOPROB=76
           NEVENT=154
           NTERM=NEVENT-1
           DO 103 J=1,NTERM
           READ(5,102)J,RAVAIL(J)
   102     FORMAT(I10,F10.2)
   103     CCNTINUE
           DO 5999 NR=LOW,NOPROB
           DO 101 I=1,NOACTS
           READ(5,100)I,ISTART(I),IEND(I),COST(I),OTCCST(I),
          1MANDAY(I)
   100     FORMAT(3I10,2F10.2,I10)
   101     CONTINUE
           DO 104 I=1,NOACTS
           SIJ(I)=0.0
           XIJ(I)=0.0
   104     CCNTINUE
           CALL DURMIN(NEVENT,NOACTS)
           IF(TIME(NEVENT).LE.TARGET)GO TO 105
           CALL COSTMN(NOACTS,PCOST,BOUND)
           CALL DURMIN(NEVENT,NOACTS)
           IF(TIME(NEVENT).LT.TARGET)GO TO 105
           GO TO 1C9
   105     CONTINUE
           DC 106 I=1,NOACTS
           IF(SIJ(I).GT.0.0)GO TO 107
   106     CONTINUE
           GC TO 109
   107     CONTINUE
           IF(TIME(NEVENT).GE.TARGET)GO TO 1C9
   108     CALL REDUCT(NOACTS,TARGET,NEVENT)
           CALL DURMIN(NEVENT,NOACTS)
           IF(TIME(NEVENT).GE.TARGET)GO TO 109
           GC TO 105
   109     WRITE(6,110)NR
   110     FCRMAT('1',T21,'PRCBLEM NUMBER ',I2,/)
           CALL OUTPUT(NOACTS,PCOST,NEVENT,TARGET)
  5999     CONTINUE
           STCP
           END


           SUBROUTINE COSTMN(NOACTS,PCOST,BOUND)
C
           CCMMON ISTART(200),IEND(200),COST(200),OTCCST(200),
          1MANDAY(200),RAVAIL(200),XIJ(200),SIJ(200),IALFA(200),
          2TIME(200),XAVAIL(200)
           DIMENSION HIJ(200)
           DC 300 I=1,NOACTS
           HIJ(I)=OTCOST(I)-COST(I)-PCOST/XAVAIL(I)
   300     CONTINUE
           TCTCT=0.0
           NEND=NOACTS-1
   304     CONTINUE
           DO 301 I=1,NEND
           IF(HIJ(I+1).LE.HIJ(I))GO TO 301
```

```fortran
      SIJ(I)=MANDAY(I)
      TOTOT=TOTOT+SIJ(I)
      HIJ(I)=-999999.9
      IF(TOTOT.GT.BOUND)GO TO 302
  301 CONTINUE
      IF(TOTOT.LT.BOUND)GO TO 304
      GO TO 303
  302 CONTINUE
      SIJ(I)=MANDAY(I)-(TOTOT-BOUND)
  303 CONTINUE
      RETURN
      END


      SUBROUTINE REDUOT(NOACTS,TARGET,NEVENT)
C
      COMMON ISTART(200),IEND(200),COST(200),OTCOST(200),
     1MANDAY(200),RAVAIL(200),XIJ(200),SIJ(200),IALFA(200),
     2TIME(200),XAVAIL(200)
      DIMENSION DIJ(200)
      DO 200 I=1,NOACTS
      IF(SIJ(I).GT.0.0)GO TO 201
      DIJ(I)=-999999.9
      GO TO 200
  201 DIJ(I)=OTCOST(I)
  200 CONTINUE
      DSTAR=DIJ(1)
      MQ=1
      NNN=NOACTS-1
      DO 202 I=1,NNN
      IF(DIJ(I+1).LE.DSTAR)GO TO 202
      MQ=I+1
      DSTAR=DIJ(I+1)
  202 CONTINUE
      SIJ(MQ)=SIJ(MQ)-XAVAIL(MQ)*(TARGET-TIME(NEVENT))
      IF(SIJ(MQ).LT.0.0)SIJ(MQ)=0.0
      XIJ(MQ)=FLOAT(MANDAY(MQ))-SIJ(MQ)
      RETURN
      END


      SUBROUTINE DURMIN(NEVENT,NOACTS)
C
      COMMON ISTART(200),IEND(200),COST(200),OTCOST(200),
     1MANDAY(200),RAVAIL(200),XIJ(200),SIJ(200),IALFA(200),
     2TIME(200),XAVAIL(200)
      DO 1000 I=1,NOACTS
      Q=0.0
      II=ISTART(I)
      III=IEND(I)-1
      DO 1000 J=II,III
      IF(RAVAIL(J).LE.Q)GO TO 1000
      XIJ(I)=FLOAT(MANDAY(I))-SIJ(I)
      Q=RAVAIL(J)
      IALFA(I)=J
      XAVAIL(I)=RAVAIL(J)
 1000 CONTINUE
      TIME(1)=0.0
      NTERM=NEVENT-1
      DO 1001 J=1,NTERM
      TIME(J+1)=TIME(J)
      DO 1001 I=1,NOACTS
      IF(IALFA(I).EQ.J)TIME(J+1)=TIME(J+1)+XIJ(I)/RAVAIL(J)
 1001 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE OUTPUT(NOACTS,PCOST,NEVENT,TARGET)
C
      COMMON ISTART(200),IEND(200),COST(200),OTCOST(200),
     1MANDAY(200),RAVAIL(200),XIJ(200),SIJ(200),IALFA(200),
     2TIME(200),XAVAIL(200)
      WRITE(6,403)
      DO 400 I=1,NOACTS
      WRITE(6,402)ISTART(I),IEND(I),XIJ(I),SIJ(I)
  402 FORMAT(T11,'(',I3,',',I3,')',T31,F10.2,T51,F10.2)
  400 CONTINUE
  403 FORMAT('0',T12,'ACTIVITY',T31,'NORMAL MANDAYS',T50,
     1'OVERTIME MANDAYS',/)
      TOTC=0.0
      DO 404 I=1,NOACTS
      TOTC=TOTC+COST(I)*XIJ(I)+OTCOST(I)*SIJ(I)
  404 CONTINUE
      IF(TIME(NEVENT).GT.TARGET)TOTC = TOTC + PCOST *
     1(TIME(NEVENT)-TARGET)
      WRITE(6,405)TOTC
  405 FORMAT('0',T11,'TOTAL PROJECT COST =',F10.2)
      NEND=NEVENT-1
      DO 406 J=1,NEND
      WRITE(6,407)J,RAVAIL(J)
  406 CONTINUE
  407 FORMAT('0',T11,I3,T21,F10.2)
      WRITE(6,410)
  410 FORMAT('1',T21,'PROJECT EVENT TIMES ARE:',/)
      DO 408 J=1,NEVENT
      WRITE(6,409)J,TIME(J)
  409 FORMAT(T21,'T(',I3,') =',F10.2,'    DAYS')
  408 CONTINUE
      RETURN
      END
```

LIST OF REFERENCES

1. Balas, E., "An Additive Algorithm for Solving Linear Programs with Zero-One Variables," Operations Research, v. 13, No. 4, p. 517-546, July-August 1965.

2. Balas, E., "Finding a Minimaximal Path in a Disjunctive PERT Network," Theorie des Graphes, International Symposium, Rome 1966, Dunod, 1967.

3. Balas, E., "Project Scheduling with Resource Constraints," Section 5 in Beale, E. M. L., (Ed.), Applications of Mathematical Programming, American Elsevier, 1970.

4. Balas, E., "Machine Sequencing Via Disjunctive Graphs: An Implicit Enumeration Algorithm," Operations Research, v. 17, No. 6, p. 941-957, November-December 1969.

5. Balinski, M. L., "Integer Programming: Methods, Uses, Computation," Management Science, v. 12, No. 3, p. 253-313, November 1965.

6. Beale, E. M. L., "On Quadratic Programming," Naval Research Logistics Quarterly, v. 6, No. 3, p. 227-243, September 1959.

7. Benders, J. F., "Partitioning Procedures for Solving Mixed-Variables Programming Problems," Numerische Mathematik, v. 4, No. 3, p. 238-252, 1962.

8. Berman, E. B., "Resource Allocation in a PERT Network Under Continuous Activity Time-Cost Functions," Management Science, v. 10, No. 4, p. 734-745, July 1964.

9. Bershchanskii, Ya. M., "Optimal Distribution of Resources Among Tasks of a Complex Project," Automation and Remote Control, (English translation of Avtomatika i Telemekhanika), No. 4, p. 575-584, April 1969.

10. Bertier, P. and Roy, B., "Une Procedure de Resolution Pour une Classe de Problemes Pouvant Avoir Un Caractere Combinatoire," ICC Bulletin, v. 4, No. 1, p. 19-27, January-March 1965.

11. Bowman, E. H., "The Schedule Sequencing Problem," Operations Research, v. 7, No. 5, p. 621-624, September-October 1959.

12. Bowman, E. H., "Assembly Line Balancing by Linear Programming," Operations Research, v. 8, No. 3, p. 385-389, May-June 1960.

13. Brooks, G. H. and White, C. R., "An Algorithm for Finding Optimal or Near-Optimal Solutions to the Production Scheduling Problem," _Journal of Industrial Engineering_, v. 16, No. 1, p. 34-40, January 1965.

14. Burgess, A. R. and Killebrew, J. B., "Variation in Activity Level on a Cyclical Arrow Diagram," _Journal of Industrial Engineering_, v. 13, No. 2, p. 76-83, March-April 1962.

15. Charlton, J. M. and Death, C. C., "A Generalized Machine-Scheduling Algorithm," _Operational Research Quarterly_, v. 21, No. 1, p. 127-134, January 1970.

16. Charlton, J. M. and Death, C. C., "A Method of Solution for General Machine-Scheduling Problems," _Operations Research_, v. 18, No. 4, p. 689-707, July-August 1970.

17. Charnes, A. and Cooper, W. W., "A Network Interpretation and a Directed Sub-Dual Algorithm for Critical-Path Scheduling," _Journal of Industrial Engineering_, v. 13, No. 4, p. 213-219, July 1962.

18. Clark, C. E., "The Optimum Allocation of Resources Among the Activities of a Network," _Journal of Industrial Engineering_, v. 12, No. 1, p. 11-17, January 1961.

19. Conway, R. W., Maxwell, W. L., and Miller, L. W., _Theory of Scheduling_, Addison-Wesley, 1967.

20. Cullingford, G. and Prideaux, J. D. C. A., "A Variational Study of Optimal Resource Profiles," _Management Science_, v. 19, No. 9, p. 1067-1081, May 1973.

21. Dantzig, G. B., "A Machine-Job Scheduling Model," _Management Science_, v. 6, No. 2, p. 191-196, January 1960.

22. Dantzig, G. B. and Wolfe, P. M., "The Decomposition Principle for Linear Programs," _Operations Research_, v. 8, No. 1, p. 101-111, January-February 1960.

23. Dantzig, G. B. and Wolfe, P. M., "The Decomposition Algorithm for Linear Programming," _Econometrica_, v. 29, No. 4, p. 767-778, October 1961.

24. Dantzig, G. B., _Linear Programming and Extensions_, Princeton University Press, 1962.

25. Davis, E. W., "Resource Allocation in Project Network Models — A Survey," _Journal of Industrial Engineering_, v. 17, No. 4, p. 177-188, April 1966.

26.   Davis, E. W. and Heidorn, G. E., "An Algorithm for Optimal Project Scheduling Under Multiple Resource Constraints," _Management Science_, v. 17, No. 12, p. B803-B816, August 1971.

27.   Falk, J. E. and Horowitz, J. L., "Critical Path Problems with Concave Cost-Time Curves," _Management Science_, v. 19, No. 4, p. 446-455, December 1972 (Part I).

28.   Faulkner, H. J. and Hillestad, J., "Application of Critical Path Techniques, Dungeness 'B' Nuclear Power Station," _Applications of Critical Path Techniques_, Brennan, J. (Ed.), American Elsevier, 1968.

29.   Fisher, M. L., _Optimal Solution of Resource Constrained Network Scheduling Problems_, Technical Report No. 56, Operations Research Center, Massachusetts Institute of Technology, September 1970.

30.   Fulkerson, D. R., "A Network Flow Computation for Project Cost Curves," _Management Science_, v. 7, No. 2, p. 167-178, January 1961.

31.   Fulkerson, D. R., "Expected Critical Path Lengths in PERT Networks," _Operations Research_, v. 10, No. 6, p. 808-817, November-December 1962.

32.   Geoffrion, A. M., "Integer Programming by Implicit Enumeration and Balas' Method," _SIAM Review_, v. 9, No. 2, p. 178-190, April 1967.

33.   Geoffrion, A. M. and Marsten, R. E., "Integer Programming Algorithms: A Framework and State of the Art Survey," _Management Science_, v. 18, No. 9, p. 465-491, May 1972.

34.   Gorenstein, S., "An Algorithm for Project (Job) Sequencing with Resource Constraints," _Operations Research_, v. 20, No. 4, p. 835-850, July-August 1972.

35.   Greenberg, H. H., "A Branch-Bound Solution to the General Scheduling Problem," _Operations Research_, v. 16, No. 2, p. 353-361, March-April 1968.

36.   Gutjahr, A. L. and Nemhauser, G. L., "An Algorithm for the Line Balancing Problem," _Management Science_, v. 11, No. 2, p. 308-315, November 1964.

37.   Hadley, G., _Linear Programming_, Addison-Wesley, 1962.

38.   Hadley, G., _Nonlinear and Dynamic Programming_, Addison-Wesley, 1964.

39. Hu, T. C., _Integer Programming and Network Flows_, Addison-Wesley, 1969.

40. Ignall, E. J., "A Review of Assembly Line Balancing," _Journal of Industrial Engineering_, v. 16, No. 4, p. 244-254, July 1965.

41. Ignall, E. and Schrage, L., "Application of the Branch and Bound Technique to Some Flow-Shop Scheduling Problems," _Operations Research_, v. 13, No. 3, p. 400-412, May 1965.

42. Jewell, W. S., "Divisible and Movable Activities in Critical-Path Analysis," _Operations Research_, v. 19, No. 2, p. 323-348, March 1971.

43. Johnson, T. J. R., _An Algorithm for the Resource Constrained Scheduling Problem_, Ph. D. Thesis, Massachusetts Institute of Technology, September 1967.

44. Karush, W., _On Scheduling a Network of Activities Under Resource Restraints Over Time_, System Development Corp. Report SP-2654, November 1966.

45. Kelley, J. E., Jr. and Walker, M. R., "Critical Path Planning and Scheduling," Proceedings of the Eastern Joint Computer Conference, 1-3 December 1959, Boston, Mass., p. 160-173.

46. Kelley, J. E., Jr., "Critical-Path Planning and Scheduling: Mathematical Basis," _Operations Research_, v. 9, No. 3, p. 296-320, May-June 1961.

47. Kelley, J. E., Jr., "The Critical-Path Method: Resources Planning and Scheduling," Chapter 21 in Muth, J. F. and Thompson, G. L., (Eds.), _Industrial Scheduling_, Prentice-Hall, 1963.

48. Lamberson, L. R. and Hocking, R. R., "Optimum Time Compression in Project Scheduling," _Management Science_, v. 16, No. 10, p. B597-B606, June 1970.

49. Lambourn, S., "Resource Allocation and Multi-project Scheduling (RAMPS) — A New Tool in Planning and Control," _The Computer Journal_, v. 5, No. 4, p. 300-304, January 1963.

50. Land, A. H. and Doig, A. G., "An Automatic Method for Solving Discrete Programming Problems," _Econometrica_, v. 28, No. 3, p. 497-520, July 1960.

51. Lasdon, L. S., _Optimization Theory for Large Systems_, MacMillan, 1970.

52. Laue, H., "Efficient Methods for the Allocation of Resources in Project Networks," _Unternehmensforschung_, v. 12, No. 2, p. 133-143, 1968.

53. Lemke, C. E. and Spielberg, K., "Direct Search Algorithm for Zero-One and Mixed-Integer Programming," _Operations Research_, v. 15, No. 5, p. 892-915, September-October 1967.

54. Levy, F. K., Thompson, G. L., and Wiest, J. D., "Multiship, Multishop, Workload Smoothing Program," _Naval Research Logistics Quarterly_, v. 9, No. 1, p. 37-44, March 1962.

55. Little, J. D. C., and others, "An Algorithm for the Traveling Salesman Problem," _Operations Research_, v. 11, No. 5, p. 972-989, September-October 1963.

56. Malcolm, D. G., and others, "Applications of a Technique for Research and Development Program Evaluation," _Operations Research_, v. 7, No. 5, p. 646-669, September-October 1959.

57. Manne, A. S., "On the Job-Shop Scheduling Problem," _Operations Research_, v. 8, No. 2, p. 219-223, March-April 1960.

58. Mason, A. T. and Moodie, C. L., "A Branch and Bound Algorithm for Minimizing Cost in Project Scheduling," _Management Science_, v. 18, No. 4, p. B158-B173, December 1971 (Part I).

59. McLaren, K. G. and Buesnel, E. L., _Network Analysis in Project Management_, Cassell, 1969.

60. Mitten, L. G., "Branch and Bound Methods: General Formulation and Properties," _Operations Research_, v. 18, No. 1, p. 24-34, January-February 1970.

61. Moder, J. J. and Phillips, C. R., _Project Management with CPM and PERT_, 2d. Ed., Reinhold Publishing Co., 1970.

62. Moodie, C. L. and Mandeville, D. E., "Project Resource Balancing by Assembly Line Balancing Techniques," _Journal of Industrial Engineering_, v. 17, No. 7, p. 377-383, July 1966.

63. Naval Ship Systems Command, Computer Applications Support and Development Office, _NAVSHIPS-MIS: Users Handbook for U. S. Naval Shipyards_, NAVSHIPS 0900-015-9010, April 1967.

64. Nemhauser, G. L., _Introduction to Dynamic Programming_, John Wiley, 1966.

65.   Nikonov, V. L. and Pluzhnikov, L. N., "Constructing the Cost Curve of a Project with Restrictions on the Resources," _Engineering Cybernetics_, (English translation of _Tekhnicheskaya Kibernetika_), No. 1, p. 38-46, January-February 1968.

66.   Parikh, S. C. and Jewell, W. S., "Decomposition of Project Networks," _Management Science_, v. 11, No. 3, p. 444-459, January 1965.

67.   Patton, G. T., _Optimal Scheduling of Resource Constrained Projects_, Ph. D. Thesis, Stanford Univeristy, August 1968.

68.   Petrovic, R., "Optimization of Resource Allocation in Project Planning," _Operations Research_, v. 16, No. 3, p. 559-568, May-June 1968.

69.   Pritsker, A. A. B., Watters, L. J., and Wolfe, P. M., "Multiproject Scheduling with Limited Resources: A Zero-One Programming Approach," _Management Science_, v. 16, No. 1, p. 93-108, September 1969.

70.   Raimond, J. -F., "Minimaximal Paths in Disjunctive Graphs by Direct Search," _IBM Journal of Research and Development_, July 1969, p. 391-399.

71.   Raimond, J. -F., "Disjunctive PERT Networks: Minimization of the Length of a Critical Path," in Lombaers, H. J. M., (Ed.), _Project Planning by Network Analysis_, North-Holland Publishing Co., 1969.

72.   Razumikhin, B. S., "The Problem of Optimal Resource Allocation," _Automation and Remote Control_, v. 26, No. 7, p. 1216-1232, July 1965.

73.   Razumikhin, B. S., "Problem of Optimal Distribution of Resources," _Automation and Remote Control_, No. 1, p. 55-66, January 1967.

74.   Razumikhin, B. S. and Stavrakov, N. E., The Problem of Distributing Resources to Minimize the Time of Execution of a Jobs Complex," _Automation and Remote Control_, No. 9, p. 1354-1365, September 1967.

75.   Rosen, J. B., "The Gradient Projection Method for Nonlinear Programming: Part I — Linear Constraints," _SIAM Journal_, v. 8, No. 1, p. 181-217, March 1960.

76.   Schrage, L., "Solving Resource-Constrained Network Problems by Implicit Enumeration — Nonpreemptive Case," _Operations Research_, v. 18, No. 2, p. 263, March-April 1970.

77.   Schrage, L., "Solving Resource-Constrained Network Problems by Implicit Enumeration — Preemptive Case," Operations Research, v. 20, No. 3, p. 668-677, May-June 1972.

78.   Schrage, L. and Woiler, S., A General Structure for Implicit Enumeration, Report, Department of Industrial Engineering, Stanford University, August 1967.

79.   Siemans, N., "A Simple CPM Time-Cost Tradeoff Algorithm," Management Science, v. 17, No. 6, p. B354-B363, February 1971.

80.   Story, A. E. and Wagner, H. M., "Computational Experience with Integer Programming for Job-Shop Scheduling," Chapter 14 in Muth, J. F. and Thompson, G. L., Industrial Scheduling, Prentice-Hall, 1963.

81.   Voronov, A. A. and Petrushinin, E. P., "Solving the Problem of Optimal Allocation of Resources by the Method of Quadratic Programming," Automation and Remote Control, v. 27, No. 11, p. 1921-1934, November 1966.

82.   Wagner, H. M., "An Integer Linear Programming Model for Machine Scheduling," Naval Research Logistics Quarterly, v. 6, No. 2, p. 131-140, June 1959.

83.   Wiest, J. D., "Some Properties of Schedules for Large Projects with Limited Resources," Operations Research, v. 12, No. 3, p. 395-418, May-June 1964.

84.   Wiest, J. D., "A Heuristic Model for Scheduling Large Projects with Limited Resources," Management Science, v. 13, No. 6, p. B359-B377, February 1967.

85.   Wiest, J. D. and Levy, F. K., A Management Guide to PERT/CPM, Prentice-Hall, 1969.

INITIAL DISTRIBUTION LIST

No. Copies

1. Defense Documentation Center   2
   Cameron Station
   Alexandria, Virginia   22314

2. Library, Code 0212   2
   Naval Postgraduate School
   Monterey, California   93940

3. Assoc. Prof. W. M. Raike, Code 55 Rj   1
   Department of Operations Research
     and Administrative Sciences
   Naval Postgraduate School
   Monterey, California   93940

4. LCDR Norman J. Shackelton, Jr. USN   3
   Prospective Repair Officer
   USS Vulcan (AR-5)
   Fleet Post Office
   New York, New York   09501

5. Naval Postgraduate School   1
   Department of Operations Research
     and Administrative Sciences
   Naval Postgraduate School
   Monterey, California   93940

6. Chief of Naval Personnel   1
   Pers 11b
   Department of the Navy
   Washington, D. C.   20370

7. Assoc. Prof. A. W. McMasters, Code 55 Mg   1
   Department of Operations Research
     and Administrative Sciences
   Naval Postgraduate School
   Monterey, California   93940

8. Disting. Prof. F. D. Faulkner, Code 53 Fa   1
   Department of Mathematics
   Naval Postgraduate School
   Monterey, California   93940

9. Professor R. R. Read, Code 55 Re   1
   Department of Operations Research
     and Administrative Sciences
   Naval Postgraduate School
   Monterey, California   93940

10. Assoc. Prof. H. A. Titus, Code 52 Ts   1
     Department of Electrical Engineering
     Naval Postgraduate School
     Monterey, California 93940

11. Assoc. Prof. D. R. Barr, Code 55 Bn   1
     Department of Operations Research
        and Administrative Sciences
     Naval Postgraduate School
     Monterey, California 93940

12. Assoc. Prof. H. Zweig, Code 55 Zw   1
     Department of Operations Research
        and Administrative Sciences
     Naval Postgraduate School
     Monterey, California 93940

13. LCDR R. D. Hager USN   1
     Repair Officer
     USS Sperry (AS-12)
     Fleet Post Office
     San Francisco, California 96601

14. Dr. S. Gorenstein   1
     Thomas J. Watson Research Center
     P. O. Box 218
     Yorktown Heights, New York 10598

15. Prof. Abraham Charnes   1
     Graduate School of Business
     University of Texas at Austin
     Austin, Texas 78712

16. Prof. Roy D. Harris   1
     Graduate School of Business
     University of Texas at Austin
     Austin, Texas 78712

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Minimizing the Cost of Projects in Naval Shipyards		5. TYPE OF REPORT & PERIOD COVERED Ph.D. Thesis; September 1973
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(e) Norman John Shackelton, Jr.		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE September 1973
		13. NUMBER OF PAGES 177
14. MONITORING AGENCY NAME & ADDRESS(If different from Controlling Office) Naval Postgraduate School Monterey, California 93940		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Project Scheduling	Job Shop Scheduling
Flow Shop Scheduling	Scheduling
Network Analysis	Mixed Integer Programming
Dynamic Programming	Disjunctive Graphs

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

    This thesis is concerned with a problem of scheduling that
arises in naval shipyards as well as in many other organizations.
The problem considered is that of minimizing the total cost of a
project with limited manpower available from the various shops
and where the number of mandays to accomplish each activity in
the project is specified. Total project cost consists of normal
direct labor cost, overtime cost, and a penalty for exceeding

20.

some specified target date. It is shown that this problem
includes several other, more common scheduling problems such as
job-shop scheduling. The relationship among the various problems
is described including the use of existing solution procedures
to solve special cases of the shipyard problem. A mixed integer
model consists of several transportation problems linked by
precedence relations. An application of dynamic programming to
the single shop case of the nonlinear model results in an
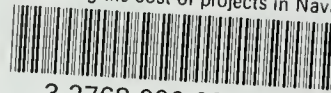efficient solution procedure.